

Figure 1 FIO System Area Network

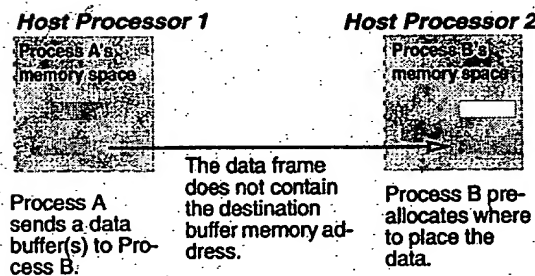


Figure 2 Data Transfer with Channel Semantics

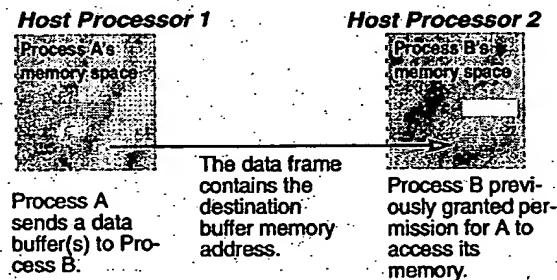


Figure 3 Data Transfer with Memory Semantics

004760"47279960

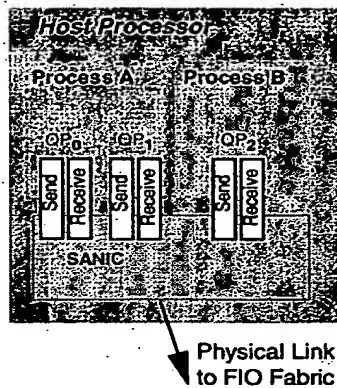


Figure 4 FIO Client Processes Communicates With FIO Hardware Through Queue Pairs

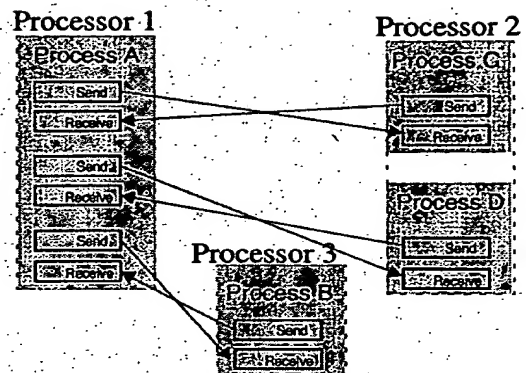


Figure 5 Connected Queue Pairs

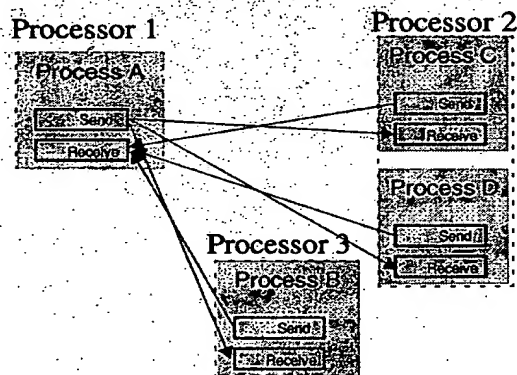


Figure 6 Connectionless Queue Pairs

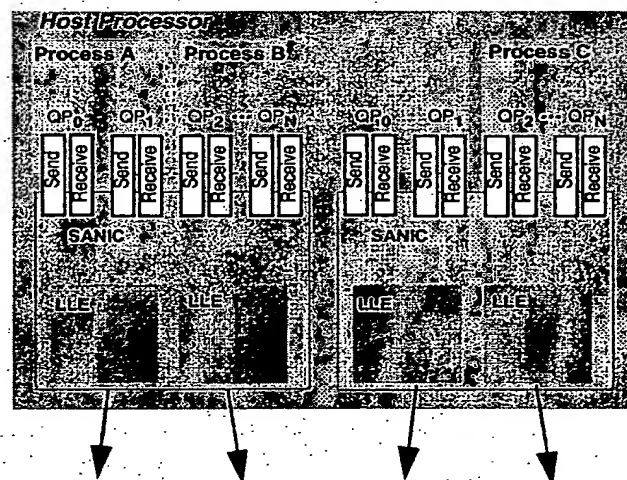
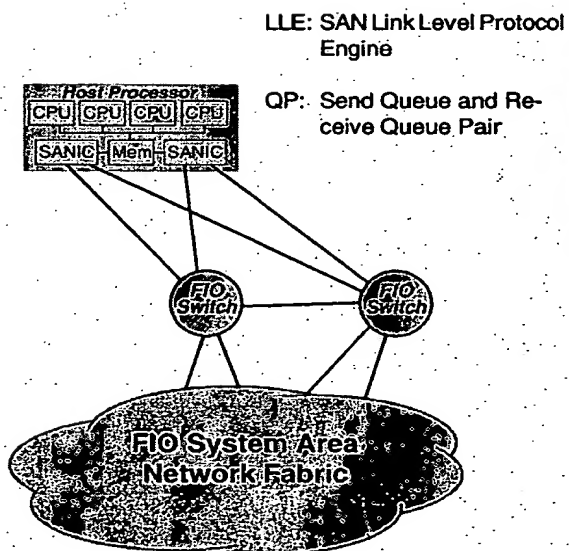


Figure 7 Multiple SANICs per host and multiple ports per SANIC

004160"41219960



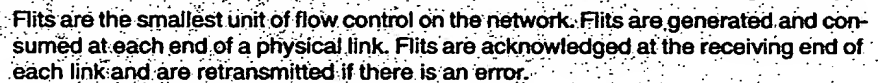
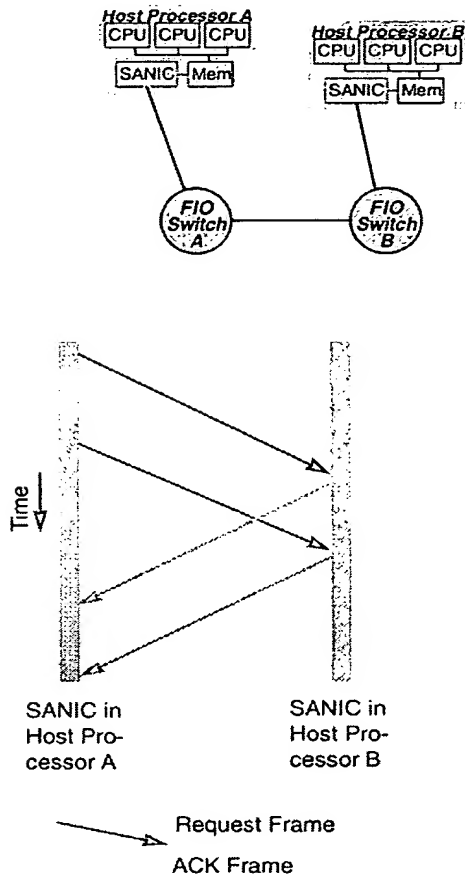


Figure 11 An FIO message partitioned into Frames and Flits



Using the message shown in Figure 11 on page 37, the Send request message is sent as two frames. Each request frame is in turn broken down into 4 flits. These ladder diagrams show the request and ACK frames going between the source and destination endnodes as well as the request and ACK flits between the source and destination of each link.

This diagram shows a message being sent with a reliable transport. Each request frame is individually acknowledged by the destination endnode.

The second part of the diagram shows the flits associated with the request and acknowledgment frames passing among the processor endnodes and the two FIO switches. An ACK frame fits inside one flit. One acknowledgment flit acknowledges several flits.

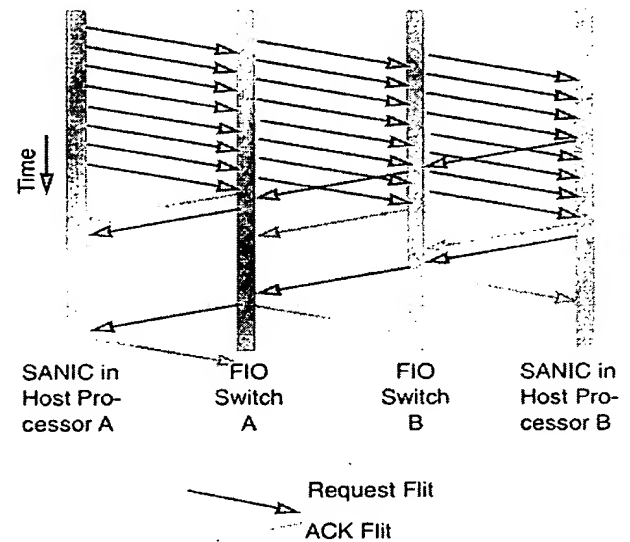


Figure 12 Multiple Request Frames (and Flits) and Their Acknowledgment Frames (and Flits)

Figure 13 Single Board Computer

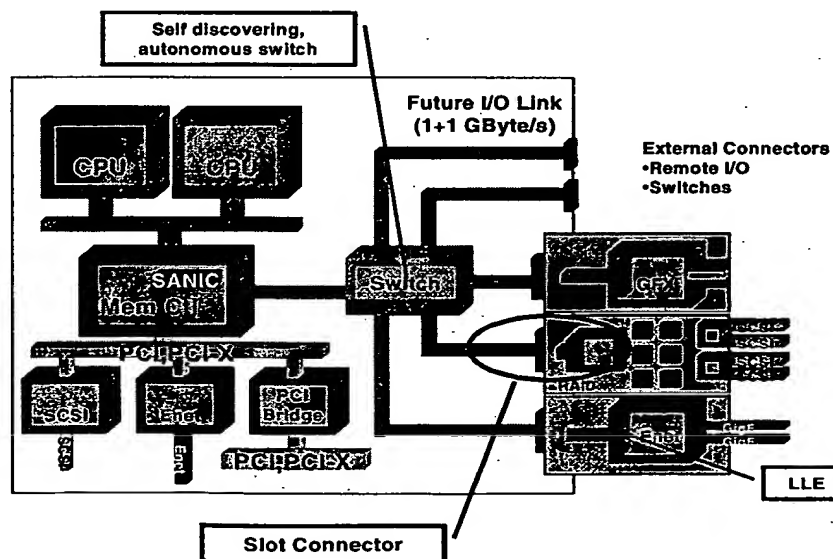


Figure 14 Remote I/O - Active Backplane

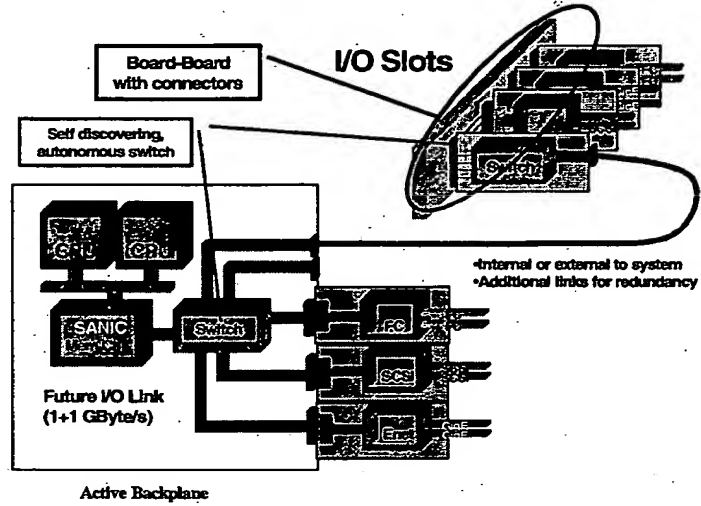


Figure 15 Remote I/O - Passive Backplane

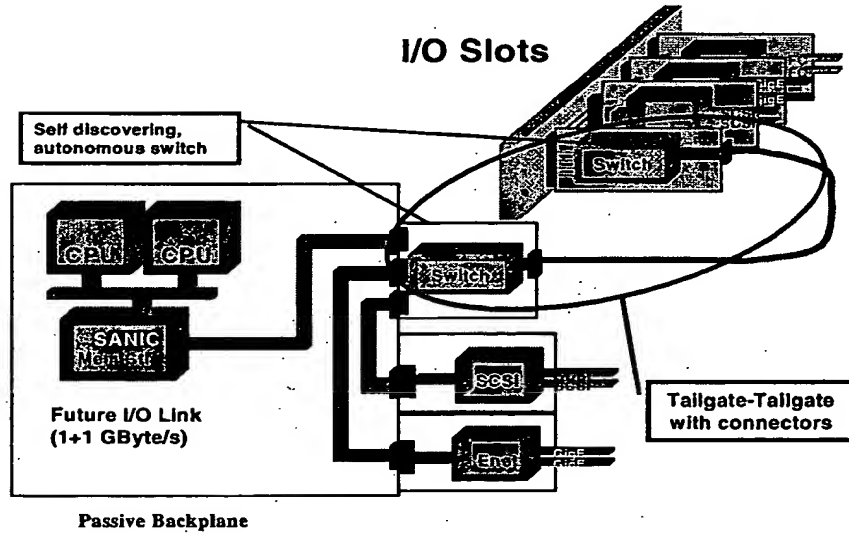


Figure 16 Chassis-to-Chassis Topology

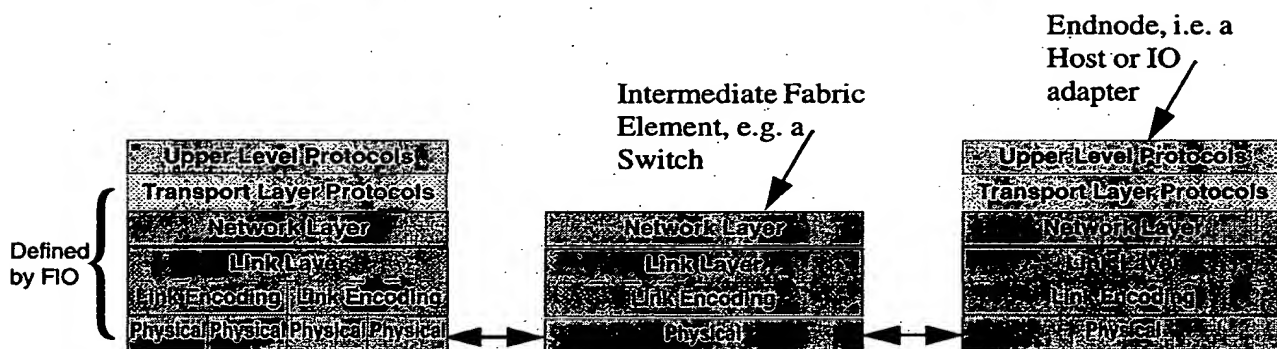
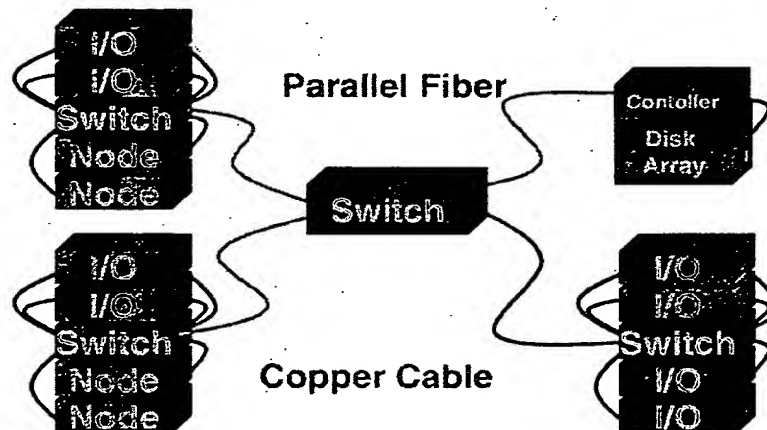


Figure 17 FIO Architecture Layers

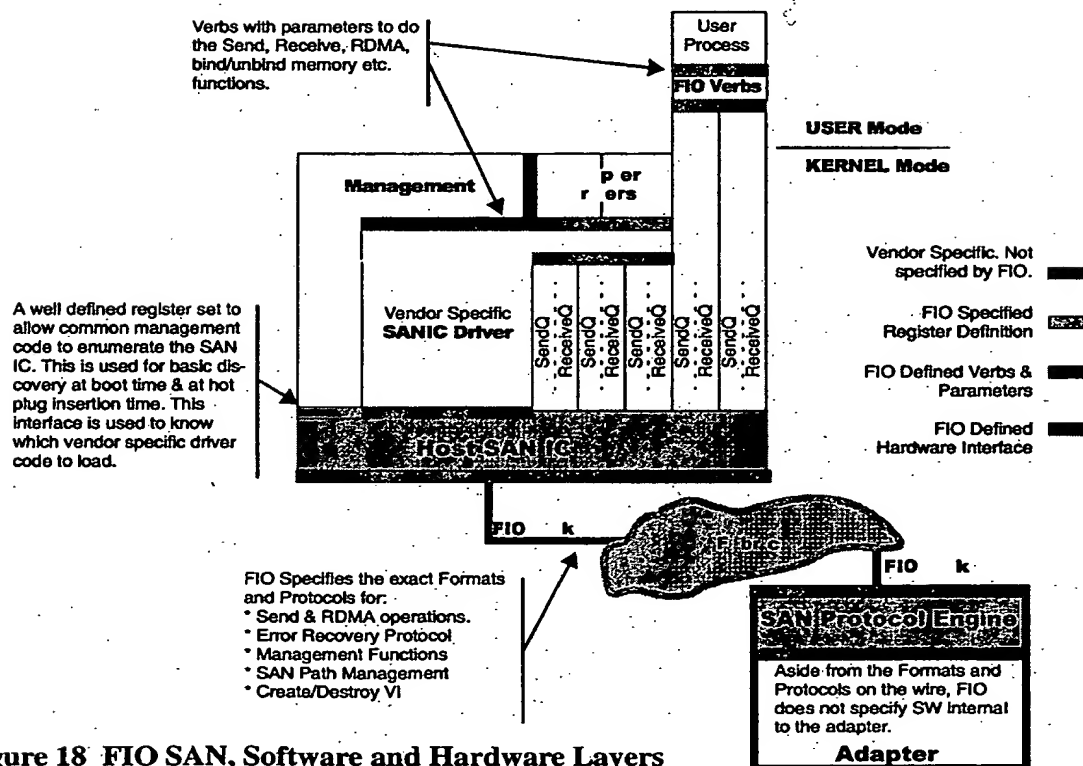


Figure 18 FIO SAN, Software and Hardware Layers

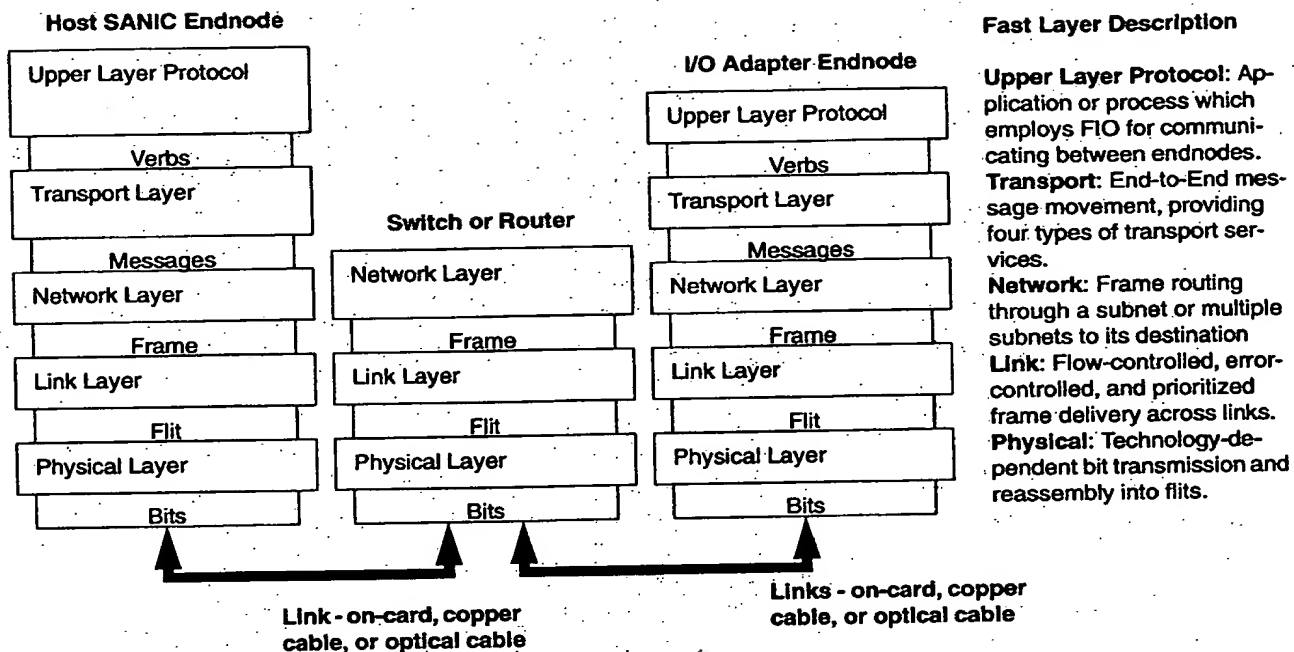


Figure 19 Future I/O Layered Architecture

Figure 20 Data flow of flit delimiters and flit body data in flit layer

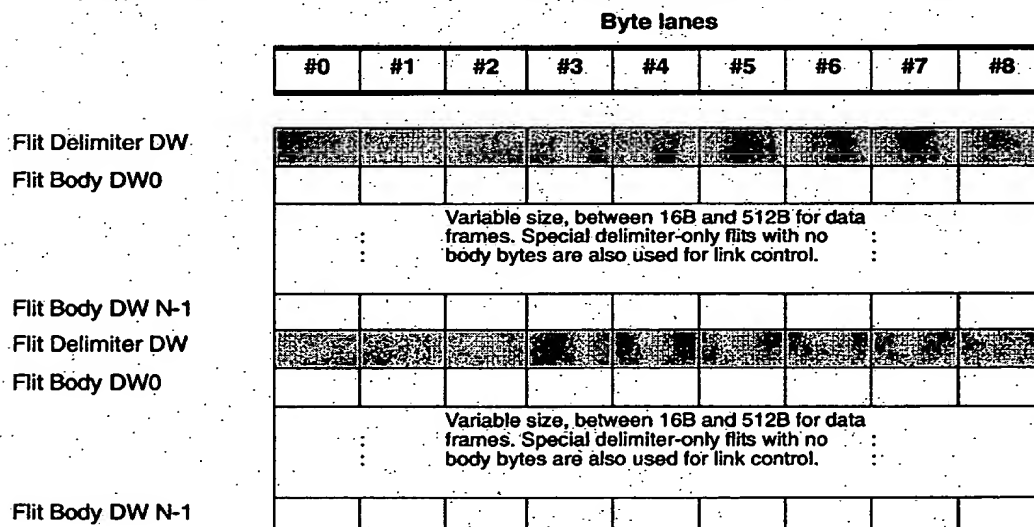


Figure 21 Flit Delimiter Fields

Byte	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TYPE			VL				SIZE						rsvd		
2	SEQ															
4	FECN		FFC				CRVL				CR					
6	LCRC															

Flit Header Field Definitions

Flit Header for following flit	TYPE (3b)	Type of following flit
	VL - Virtual Lane (4b)	Virtual Lane, in range 0-15
	SIZE (6b)	Flit size = 8 * SIZE (in bytes), SIZE = 0b00000 may either mean 0 bytes, or 512 bytes, depending on TYPE - see Figure 22 on page 55.
	SEQ - Sequence Number (16b)	Flit Sequence Number of following flit, only used for sequenced flits
Network	FECN (2b)	Forward Explicit Congestion Notification - End-to-End notification to frame destinations that the frames experienced congestion in the fabric - this information is fed back to the frame source in the ACK, to assist source injection rate control for congestion avoidance.
	FFC (4b)	Forward Flow Control - Indication, at each switch stage, of how many input ports at the preceding switches in the network have data queued for the same output port and VL. This information is aggregated through the network, so that the switch arbitration engines at following switch stages in the network can adjust policies for greater fairness across sources.
Link Flow control and error control	CRVL (4b)	Virtual Lane for which credit is being given in the CR field
	CR - Credit (4b)	Number of 64-Byte flit buffer slots that have been freed up to accept more flit body data.
	LCRC = Link-level CRC (16b)	Covers flit body of preceding flit and preceding fields of the delimiter. Uses $x^{16} + x^{12} + x^5 + 1$ CRC polynomial.

004T60"4T2T9360

Figure 22 Flit TYPE definition

TYPE	Usage	Description	SIZE	VL	SEQ	CRVL/ CR	Flit Body
0	Link Idle/Ack	Used when there are no other flits to send. Acknowledges flits received in the opposite direction across the link	SIZE=0; size is 8 bytes	15	Carries Seq number of last flit received correctly	0/15 at sender Ignored receiver	none
1	Credit-only	used to carry credit update information when there are no Frame flits to send.	Size=0; size is 8 bytes	15 - doesn't require credit	0x000- 0x7FE, increment- ing	CRVL indi- cates which VL is being given transmis- sion credit. CR indi- cates how many 64- byte flit buffer cred- its are being given.	none
2	Frame: First	Frame Flits These flits are used for transporting frames between FIO components.	Normal Flit con- tains 16*SIZE Bytes of body data. (0b00000 = 512B)	0-14 for Data Fames, 15 for Manage- ment/Ne twork Control Frames	0x000- 0x7FE, increment- ing		16B- 512B, indicated by SIZE field
3	Frame: Middle						
4	Frame: Last						
5	Frame: First and Last						
6	Init	(Initializes all flit-level parameters: VL credit to 0 on all VLs, clear all ECRC accumulators, clear SEQ to 0x000, etc.)	Size=0; size is 8 bytes	0	0x7FF unse- quenced	0/15 at sender Ignored by Receiver	none
	Pong - Link Control	Sent to acknowledge reception of a Ping flit. Not retransmitted.	Size=0; size is 8 bytes	1*	0x7FF unse- quenced		none
	Ping - Link Control	Used during link initializa- tion with the Pong flit to time the length of the link	Size=0; size is 8 bytes	2*	0x7FF unse- quenced		none
	TOD Con- trol Frame	Time-of-Day frame - Not retransmitted, since a retransmit would contain the "old" (incorrect) time.	Normal	3	0x7FF unse- quenced		16B- 512B, indicated by SIZE
		reserved -	Ignored by receiver, logged and reported as an error		4-15		
7	reserved -	Ignored by receiver, logged and reported as an error					

0961214.091400

Link Flit Logic:

Building flits from frames,
link-level flow control and
error control, etc.

Link Encoding/Decoding Logic:

link training, byte, word and
flit alignment,
encoding and decoding, fre-
quency difference compen-
sation.

Link Physical Layer:

Digital/analog conversion,
high/low-speed mux/demux,
inter-line deskew

Chip
Transmit/Receive
Interfaces

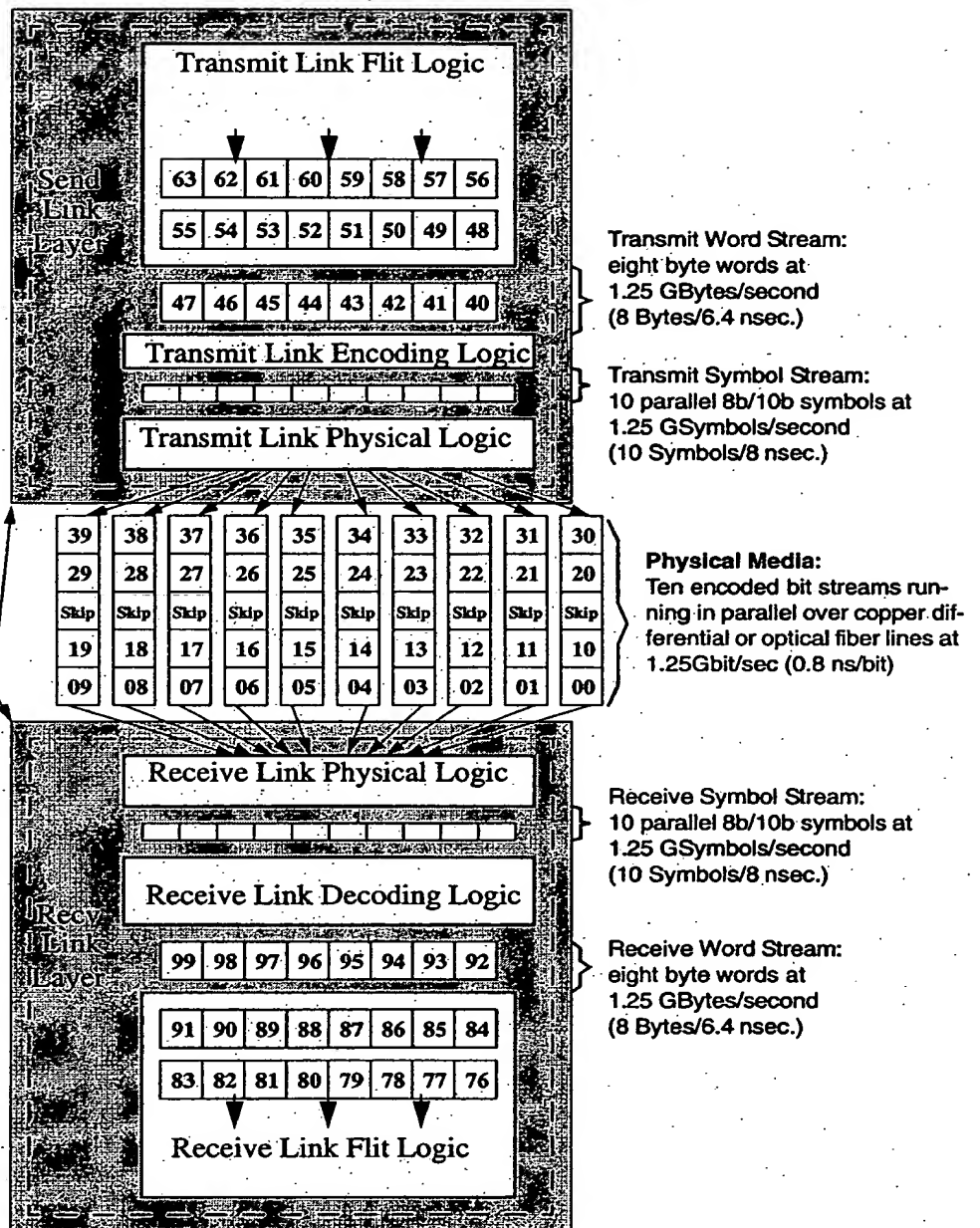


Figure 24 FIO Data Path and Interfaces for Link and Physical Layers

5B/6B coding for Data Characters		
Unencoded EDCBA	Current RD- abcdei	Current RD+ abcdei
D0: 00000 -	100111	011000
D1: 00001 -	011101	100010
D2: 00010 -	101101	010010
D3: 00011	110001	110001
D4: 00100 -	110101	001010
D5: 00101	101001	101001
D6: 00110	011001	011001
D7: 00111	111000	000111
D8: 01000 -	111001	000110
D9: 01001	100101	100101
D10:01010	010101	010101
D11:01011	110100	110100
D12:01100	001101	001101
D13:01101	101100	101100
D14:01110	011100	011100
D15:01111 -	010111	101000--
D16:10000 -	011011	100100--
D17:10001	100011	100011
D18:10010	010011	010011
D19:10011	110010	110010
D20:10100	001011	001011
D21:10101	101010	101010
D22:10110	011010	011010
D23:10111 -	111010	000101--
D24:11000	110011	001100
D25:11001 -	100110	100110--
D26:11010 -	010110	010110--
D27:11011 -	110110	001001--
D28:11100	001110	001110
D29:11101	101110	010001
D30:11110	011110	100001
D31:11111	101011	010100

3B/4B coding for Data Characters		
Unencoded HGF	Current RD- fghj	Current RD+ fghj
--.0: 000 -	1011	0100
--.1: 001	1001	1001
--.2: 010	0101	0101
--.3: 011	1100	0011
--.4: 100 -	1101	0010
--.5: 101	1010	1010
--.6: 110	0110	0110
--.7: 111 -	1110/0111	0001

5B/6B coding for Special Characters		
Unencoded EDCBA	Current RD - abcdei	Current RD + abcdei
K28:11100 -	001111	110000
K23:10111 -	111010	000101
K27:11011 -	110110	001001
K29:11101 -	101110	010001
K30:11110 -	011110	100001

3B/4B coding for Special Characters		
Unencoded HGF	Current RD - fghj	Current RD + fghj
--.0: 000 -	1011	0100
--.1: 001	0110	1001
--.2: 010	1010	0101
--.3: 011	1100	0011
--.4: 100 -	1101	0010
--.5: 101	0101	1010
--.6: 110	1001	0110
--.7: 111 -	0111	1000

Valid Special Characters		
Char Name (#)	Current RD- /Current RD+	Current RD- /Current RD+
K28.0(1C)	001111 0100--110000	1011
K28.1(3C)	001111 1001--110000	0110
K28.2(5C)	001111 0101--110000	1010
K28.3(7C)	001111 0011--110000	1100
K28.4(9C)	001111 0010--110000	1101
K28.5(BC)	001111 1010--110000	0101
K28.6(DC)	001111 0110--110000	1001
K28.7(FC)	001111 1000--110000	0111
K23.7(F7)	111010 1000--000101	0111
K27.7(FB)	110110 1000--001001	0111
K29.7(FD)	101110 1000--010001	0111
K30.7(FE)	011110 1000--100001	0111

The comma series b'0011 1110' or b'1100 0001' can be used for synchronization, since it contains a run length of 5, which can not appear in any data character or combination of data characters.

Figure 25 8b/10b Coding Conversion

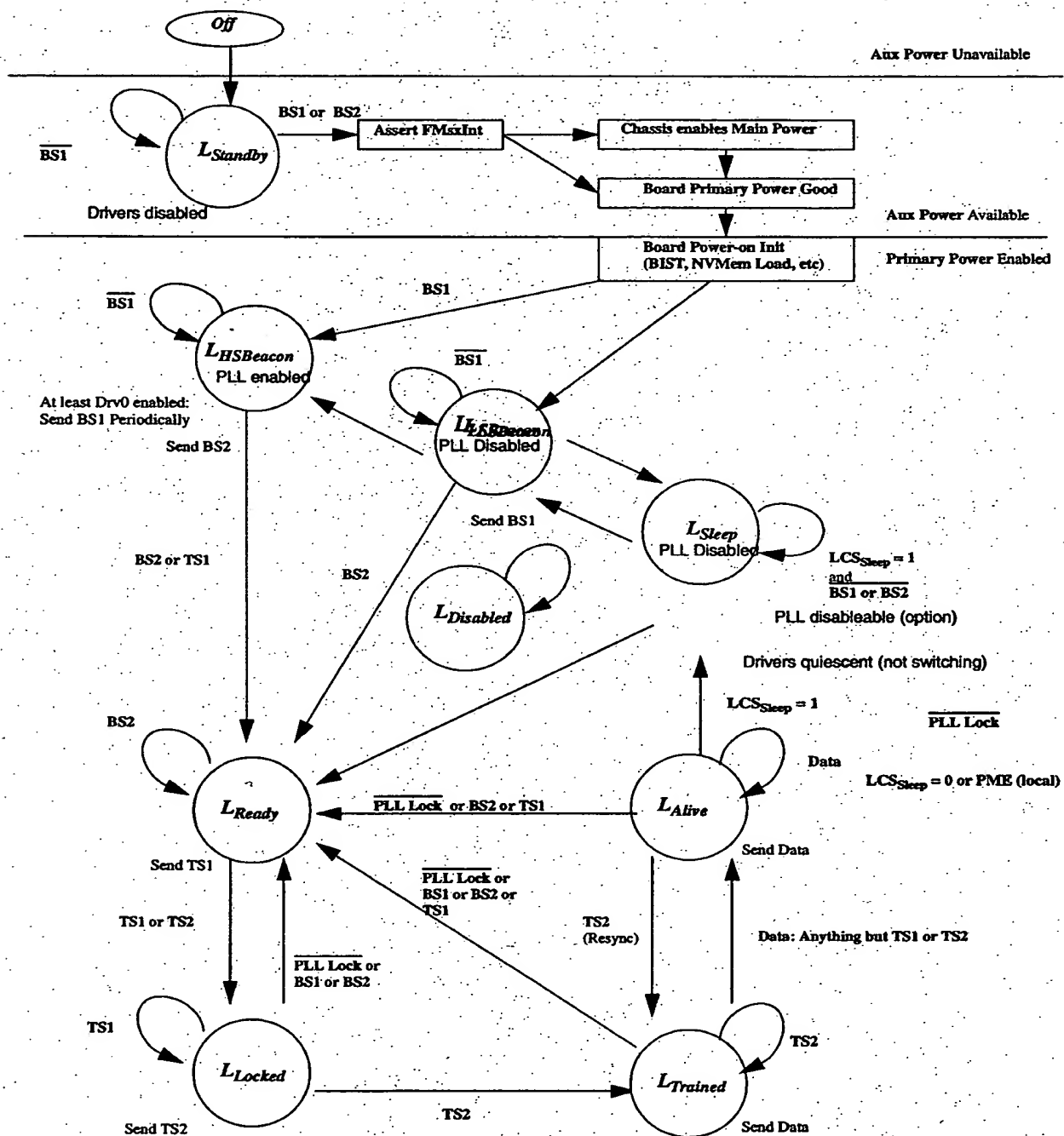


Figure 27 FIO Link Training - One End Power-on shows the typical order of transmission of beaconing and link training sequences used in bringing a link from a powered-off state to an alive and operational state.

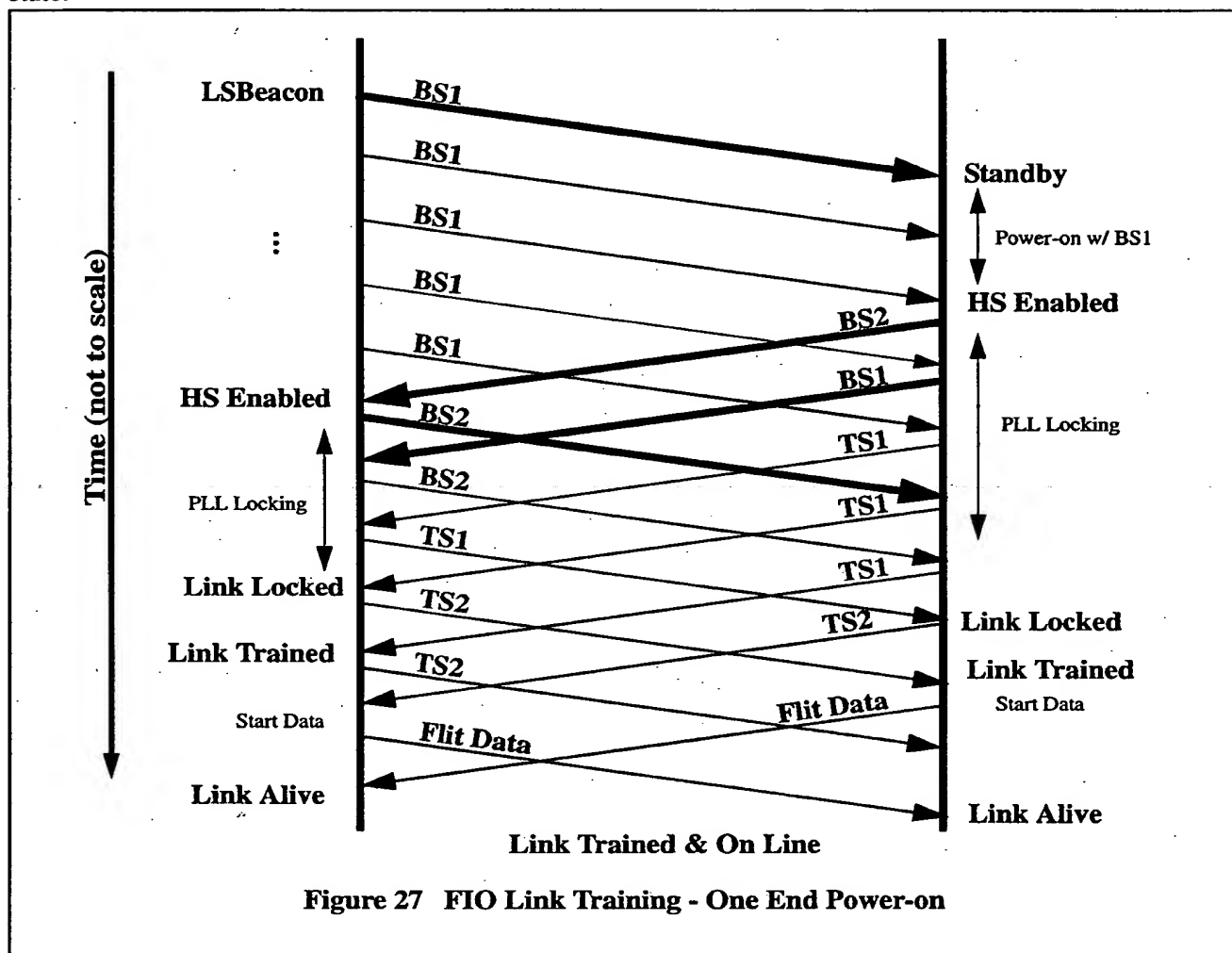


Figure 28 Future I/O Layered Architecture

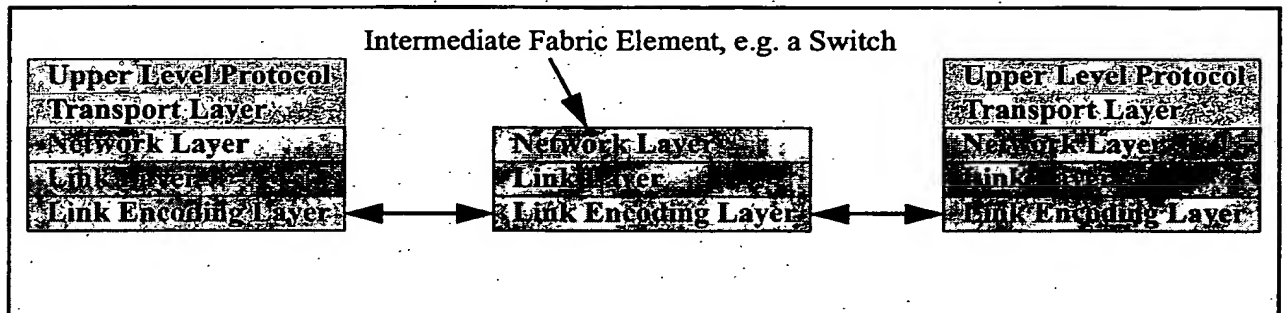


Figure 28 Future I/O Layered Architecture

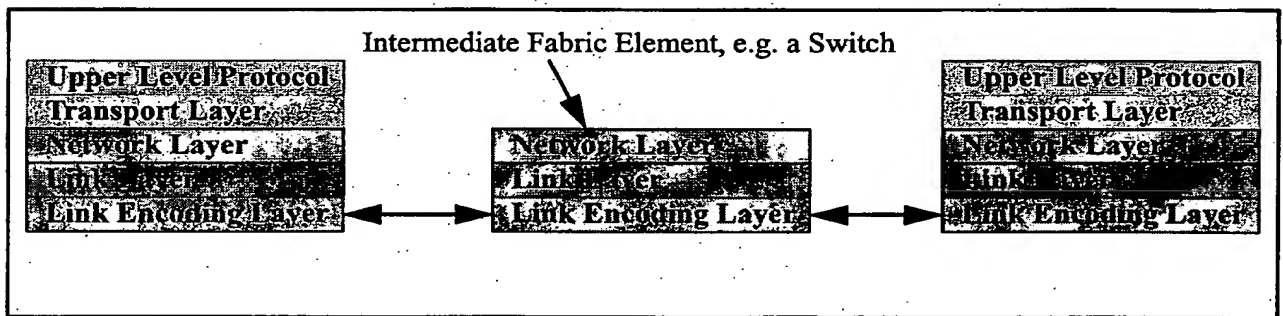
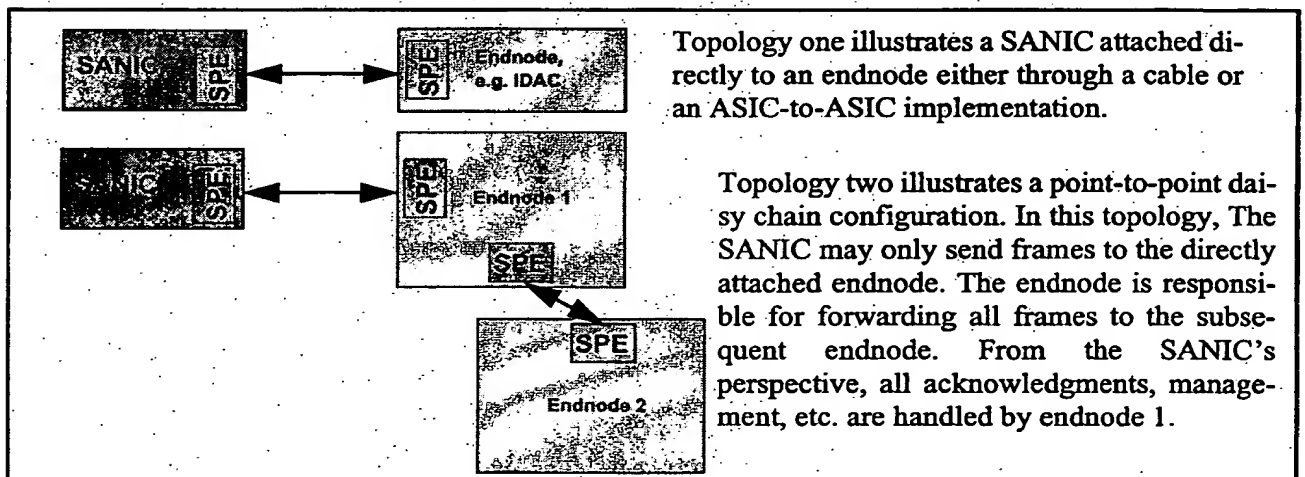


Figure 29 Sample Point-to-point Topologies



004150-121950

Figure 30 Single-board Platform

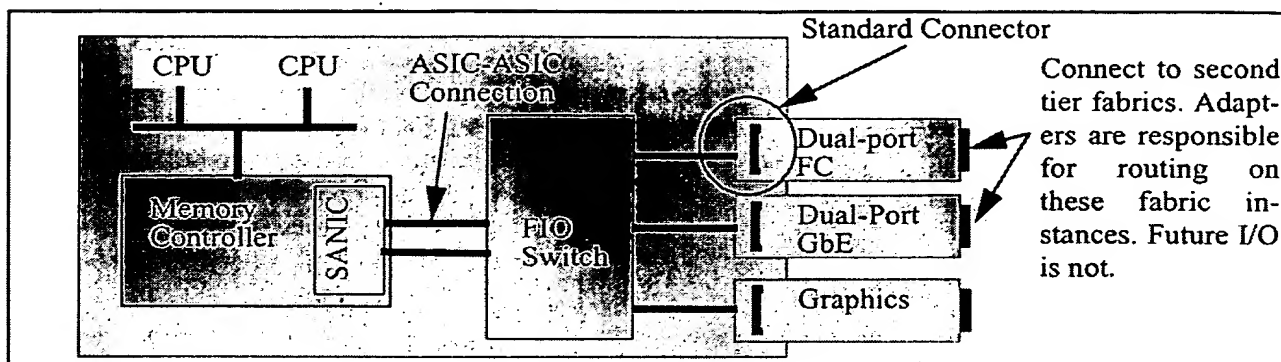


Figure 31 Passive Backplane Platform

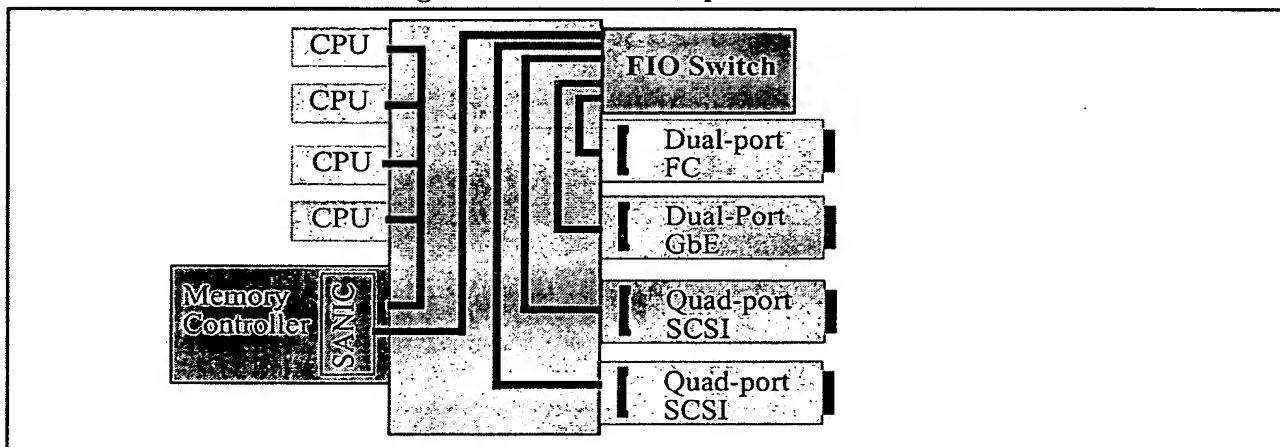


Figure 32 Platform-to-Chassis Topology

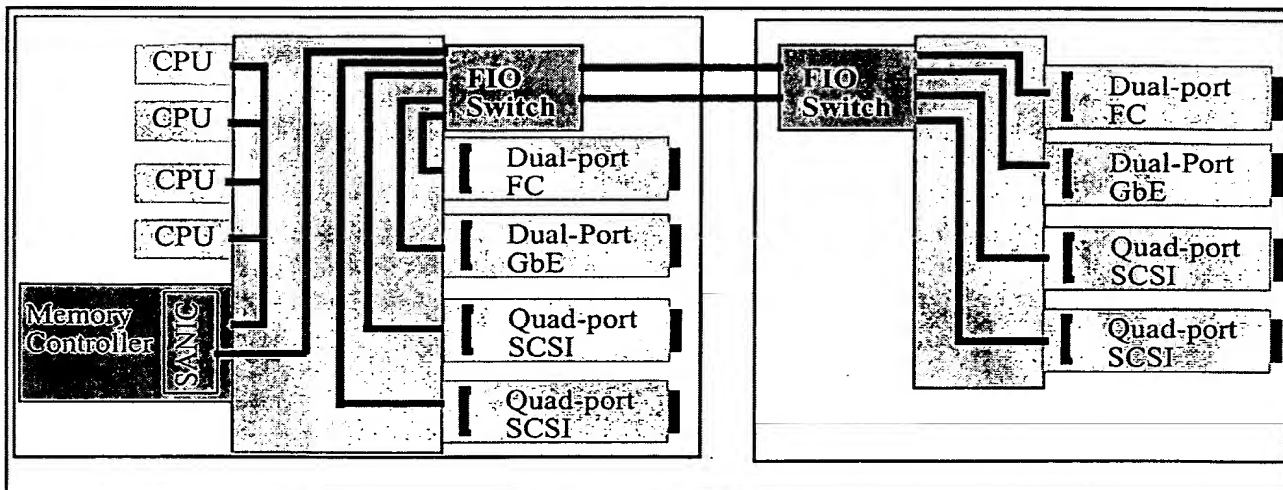


Figure 33 endnodes Connected via External Switch Elements

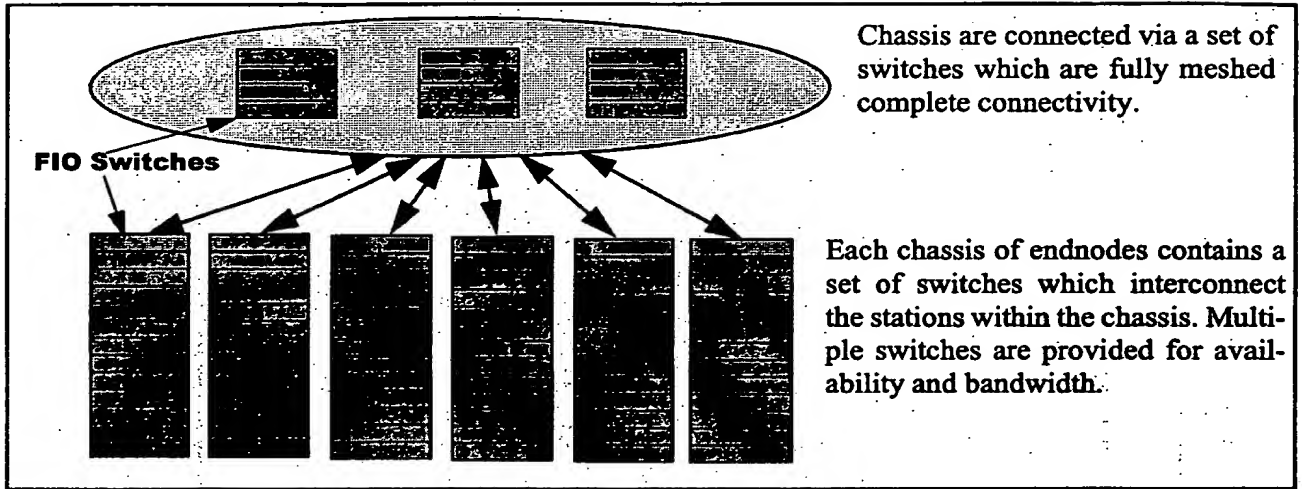


Figure 34 Leaf End-station with Embedded Switch

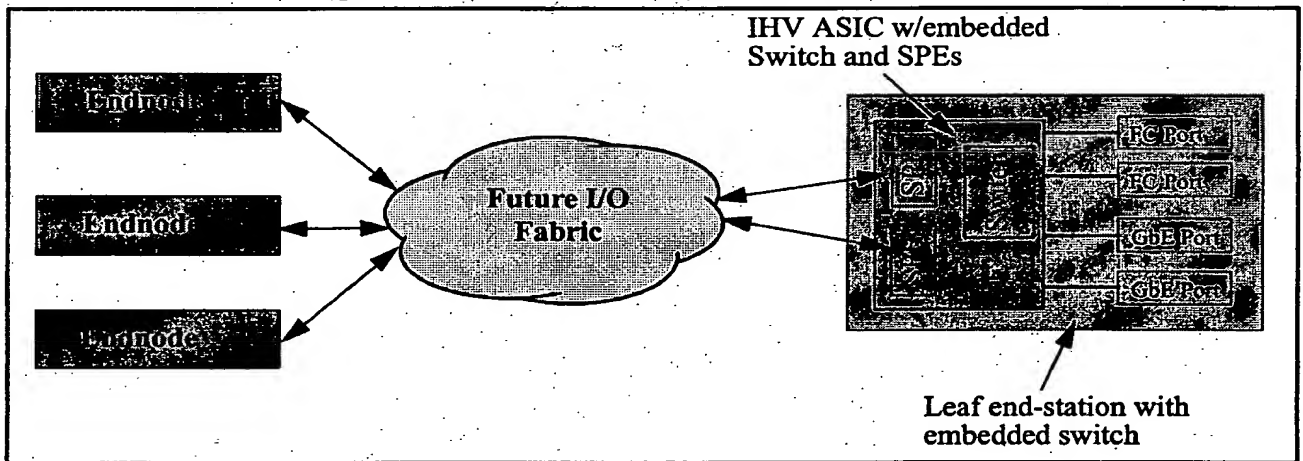


Figure 35 Sample Switch Frame Routing

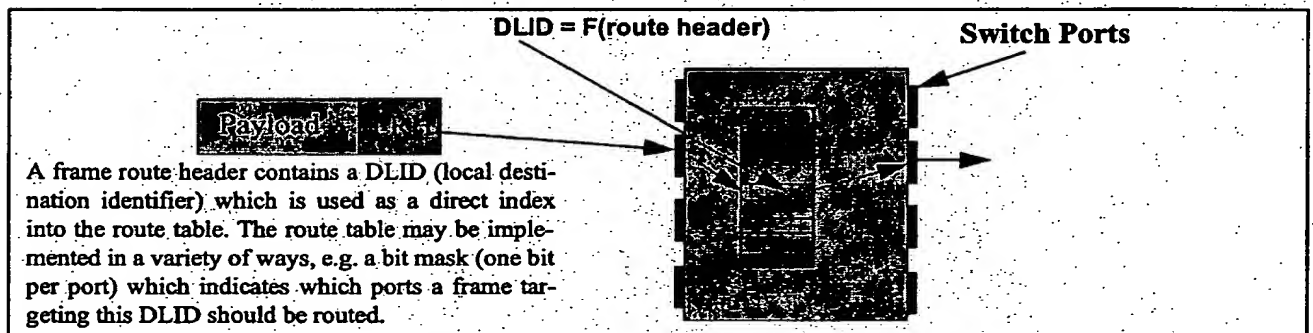


Figure 36 Sample Router Route Operation

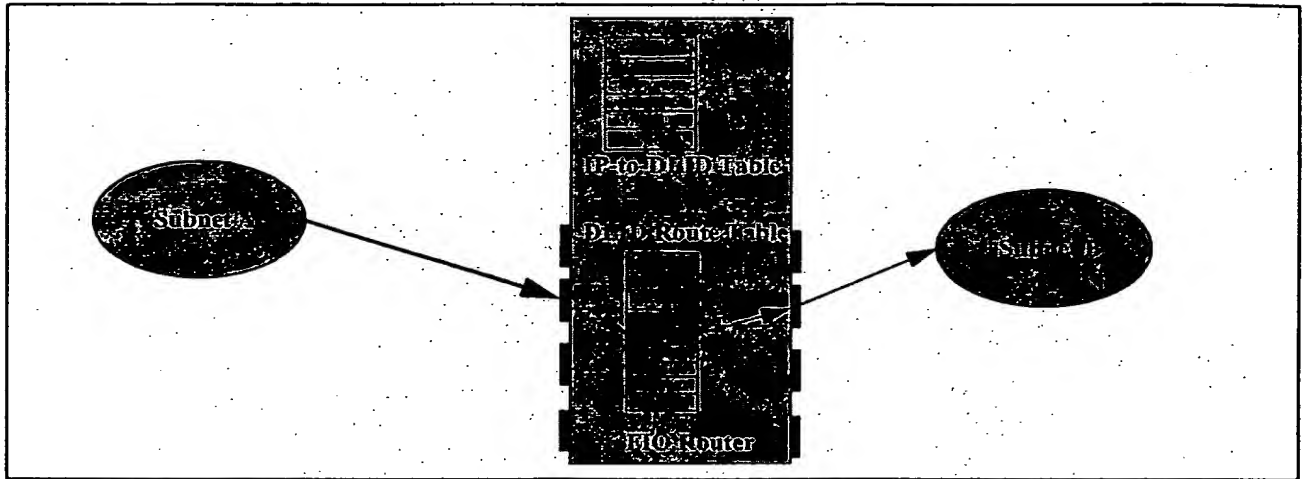


Figure 38 Graphical View of Route Headers and Payloads

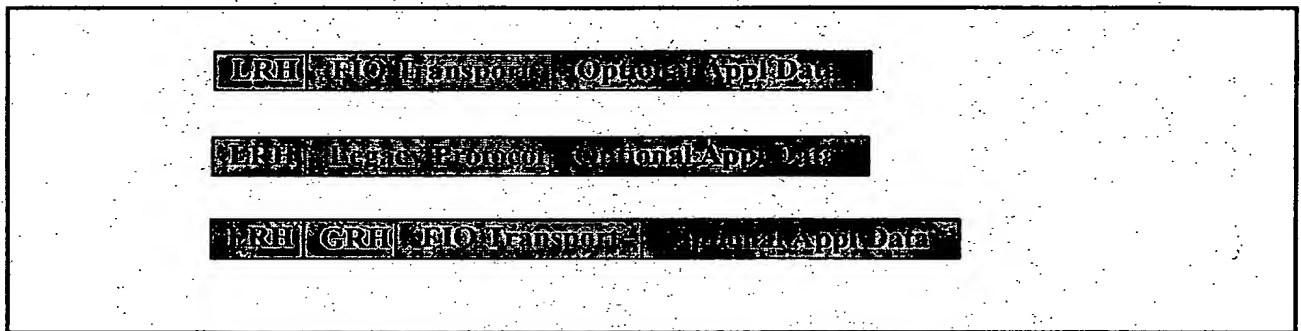
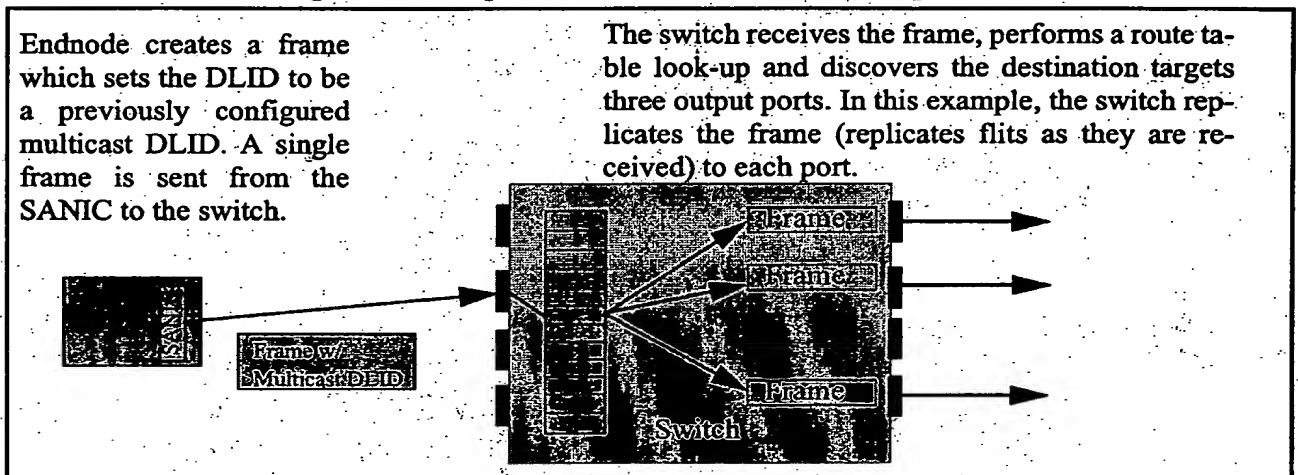


Figure 39 Sample Unreliable Multicast Send Operation



00461214-091400

Figure 37 Attribute Comparison of Switch Routing Technologies

Attribute	Wormhole	Cut-through	Store-Forward
Fabric Efficiency - Small radius configurations have essentially equivalent efficiency regardless of the algorithm used.	Medium to large radius fabrics have maximum 60% efficiency (random packet distribution). Typically, 40-50% is reality.	Due to additional memory to deal with head-of-line blocking, this algorithm provides better efficiency than wormhole.	Depending upon the amount of memory within the switch and the traffic patterns, this algorithm provides better efficiency than cut-through or wormhole.
Design Simplicity	Simplest design - does not require routing table - depends upon whether one is using source or destination routing.	Additional complexity due to additional packet buffering. Possible to implement some QoS within the switch should congestion occur else operation is essentially wormhole.	Most complex of the three but the impact varies depending the amount of QoS, memory component integration, etc.
Adaptive Routing	Partial adaptivity (based on redundant cables)	Partial adaptivity - similar to wormhole	Yes. Implementation trade-off in terms of management, ordering requirements (e.g. deals with ghost I/O), etc.
Deadlock Avoidance - no loops within the fabric allowed.	Yes, under assumption end node makes forward progress.	Similar to wormhole.	Typically use 802.1D/OSPF to avoid loops. Layer 3 switches may also modify packet header hop/TTL.
Cost, i.e. price/performance	Lowest	Higher than Wormhole but less than store-forward	Highest relative cost of the three due to increased resource management.
Memory Reqs - Each requires minimally 2X RTT slack buffers which may be either implemented in on-chip or off-chip memory	Low - Slack buffers and optional route table may be implemented on-chip	Typically additional off-chip memory for congestion buffers	Off-chip memory - amount varies with bandwidth, number of ports, possibly fabric radius.
Flow-control Reqs	Symbols (dominant) / Credit	Symbols/Credits	Symbols/Credit/None/etc.
Misc. Resource Reqs	Minimal management. Depends upon whether source or destination routing. Destination reqs external service frames to program route table.	Similar to wormhole.	Depends upon what value-added capabilities are provided. For example, if a layer 3 or layer 4 switch, then additional management tools/resources required to provide policies.
QoS Capabilities	Requires VC approach which is a limited resource and hence limited fabric QoS capabilities	Similar to wormhole.	Complex QoS capabilities. May use variety of tag or flow schemes to determine priority of service and guarantee bandwidth
Latency	Best	Same as wormhole except when congestion occurs	Complexity of providing store-forward buffer management, QoS, etc. increases latency compared to /wormhole
Fragmentation/Re-assembly	Unlimited frame size but often implemented with max MTU. FIO specifies MTUs for flits and frames.	Due to limited congestion buffering, requires max MTU. FIO specifies MTUs for flits and frames.	Max MTU to avoid skewing traffic. FIO specifies MTUs for flits and frames.

09661214 091400

Figure 40 Policy Based Routing Example

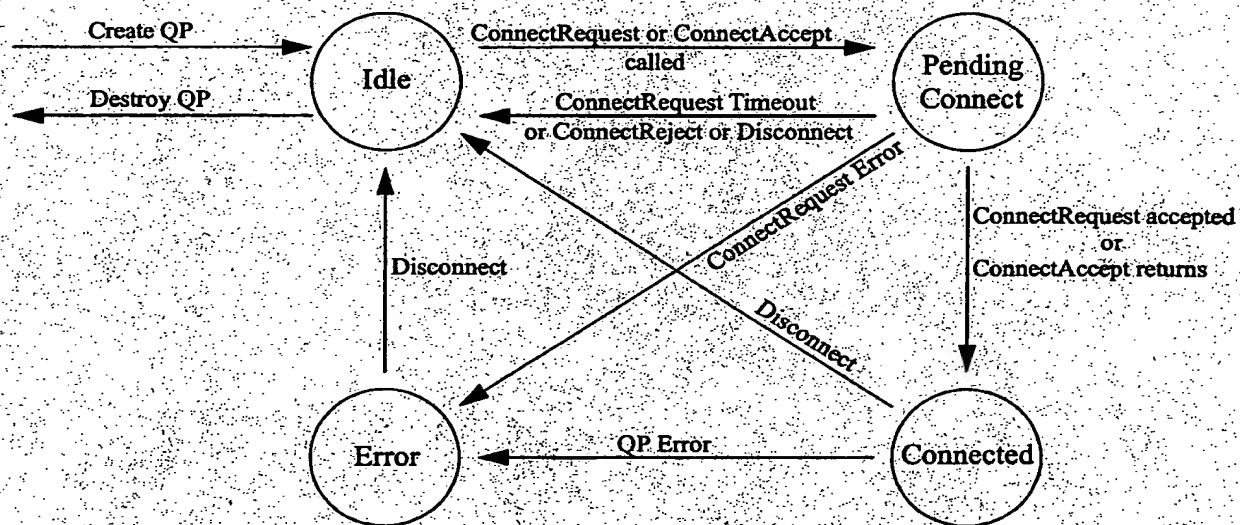
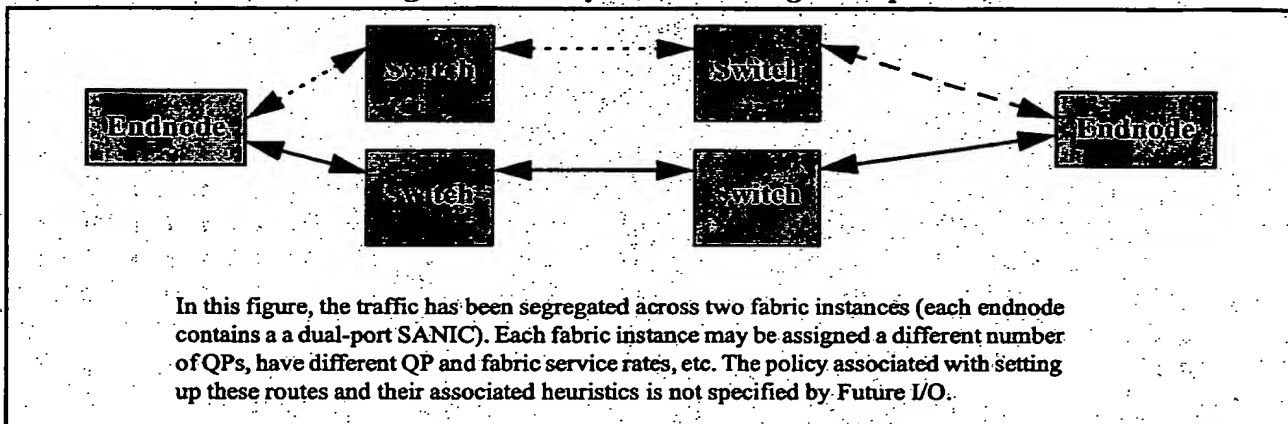


Figure 44 QP State Diagram

004T50"42T960

This figure shows two views of connected, reliable transfer QPs. Process A on Processor 1 communicates with three processes: processes C and D on Processor 2 and process E on processor 3.

The upper view shows how software might view the connection. Buffers in the Send Q flow into buffers in the receive queue on the connected QP.

The lower view gives a hardware centric view, showing some of the state maintained per connected QP.

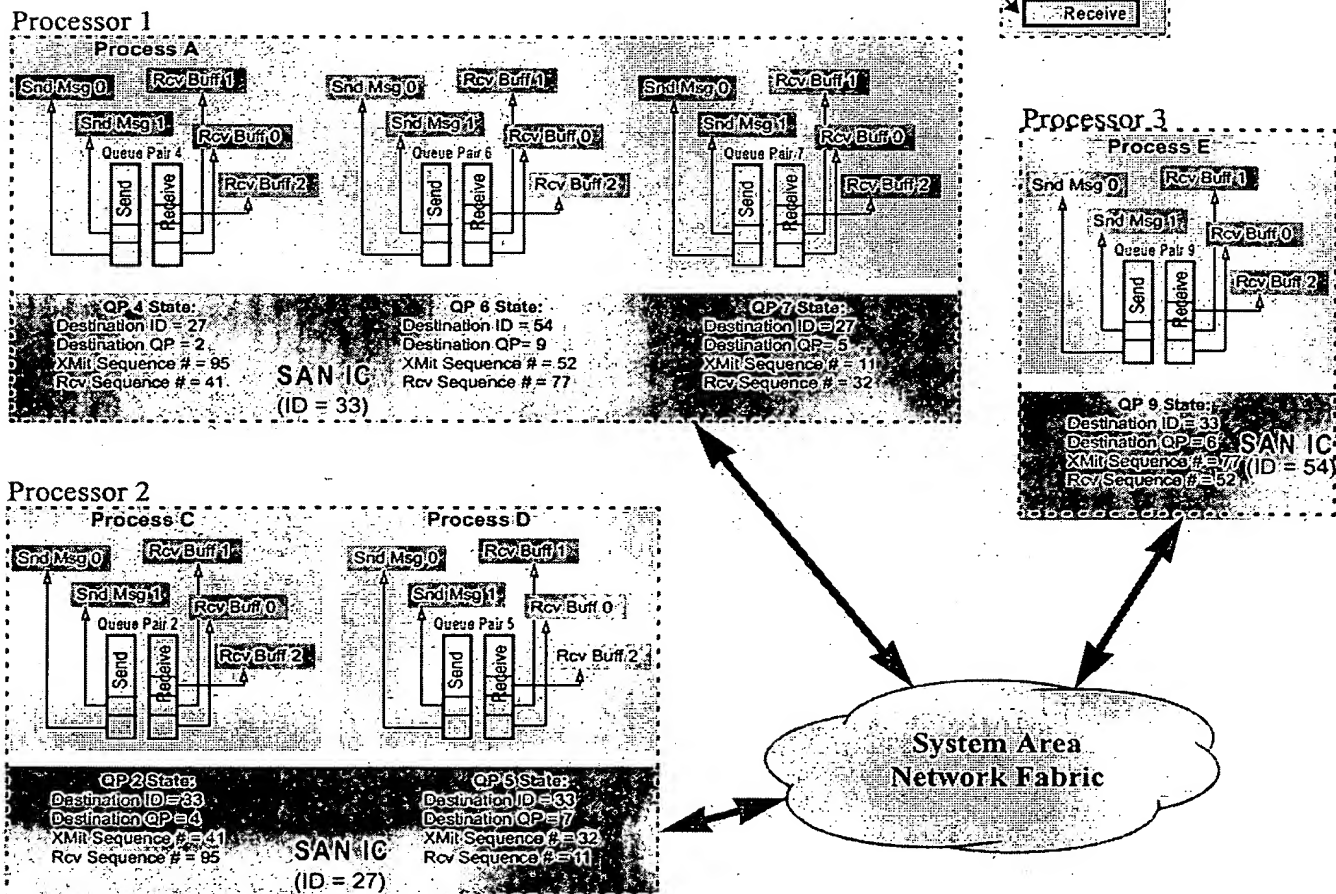
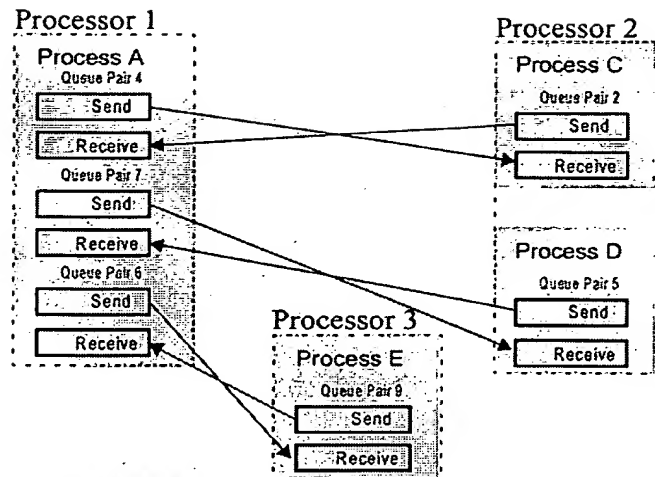
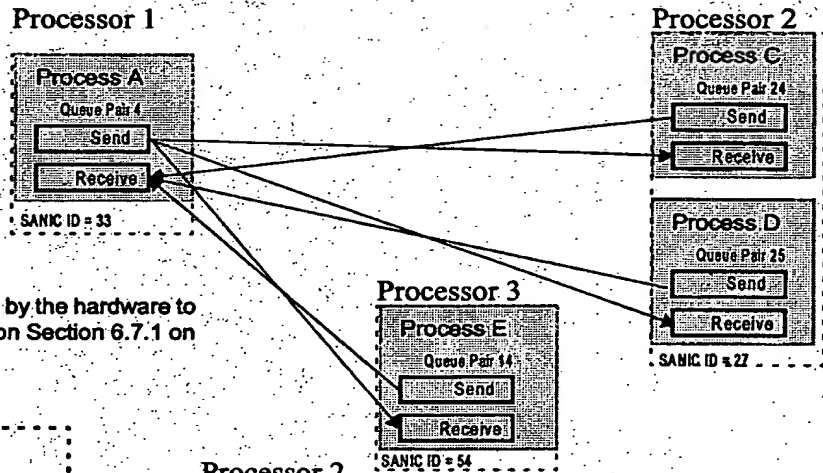


Figure 41 Connected QPs for Reliable Transfer Operation

Two views of connectionless, Reliable Datagram service. The upper figure shows a software view of Reliable Datagram communication among 4 processes on 3 processors. In this example, there is no communication among process E and processes C and D, otherwise, each QP can send to and receive from all the other queue pairs.

The lower figure shows the multiple queue pairs used by the hardware to synthesize the Reliable Datagram service. See section 6.7.1 on page 167 for more explanation of how this works.



Processor 1

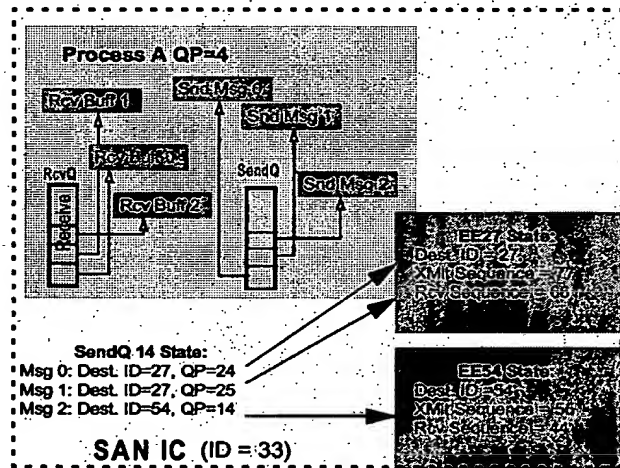


Figure 42 Connectionless QPs for Reliable Datagram

This figure shows two views of connectionless, unReliable Datagram QPs. Process A on Processor 1 communicates with three processes: processes C and D on Processor 2 and process E on processor 3.

The upper view shows how software might view the connection. Buffers in the Send Q flow into buffers in the receive queue on the connected QP.

The lower view gives a hardware centric view, showing the state maintained per connected QP.

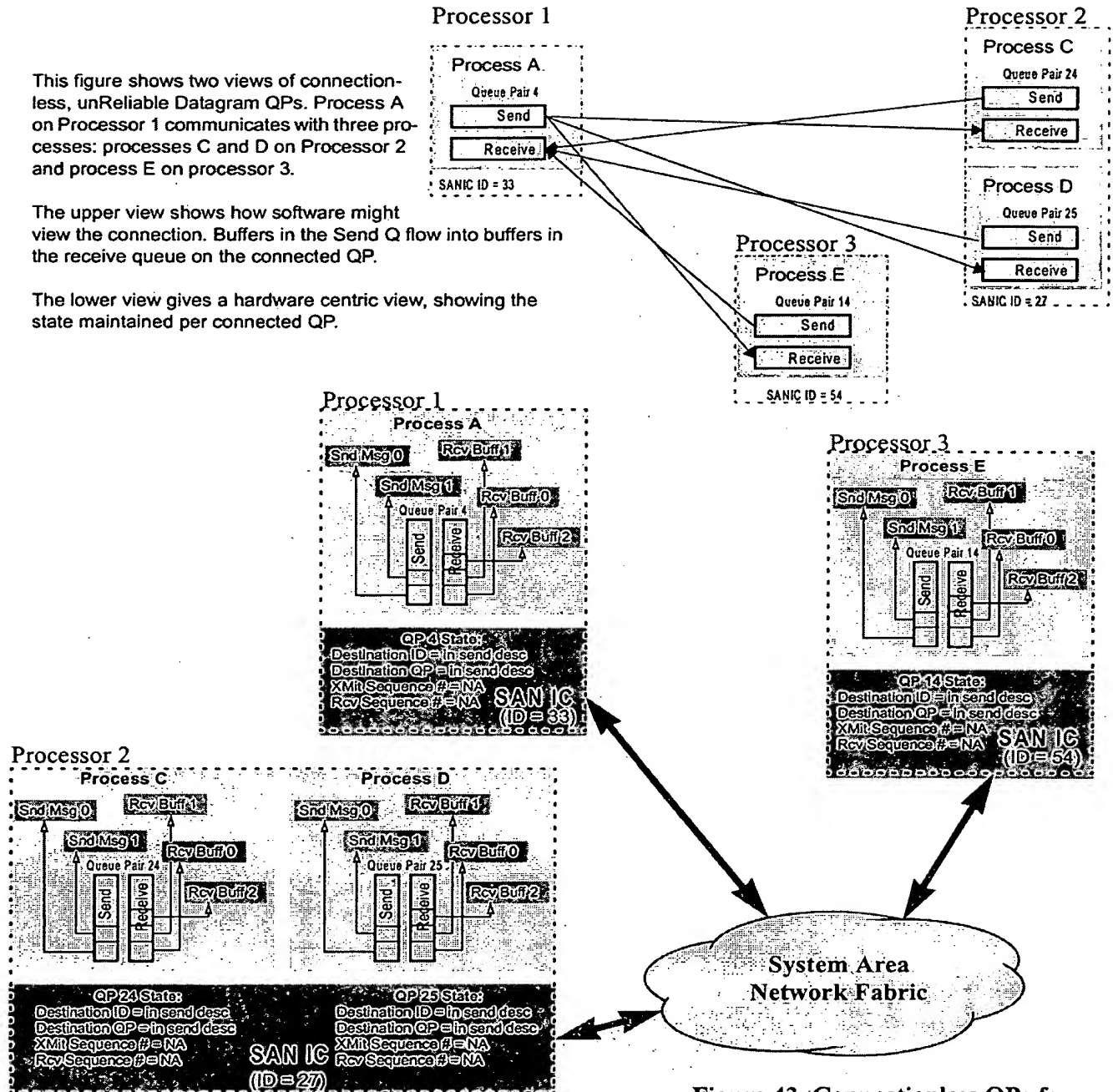
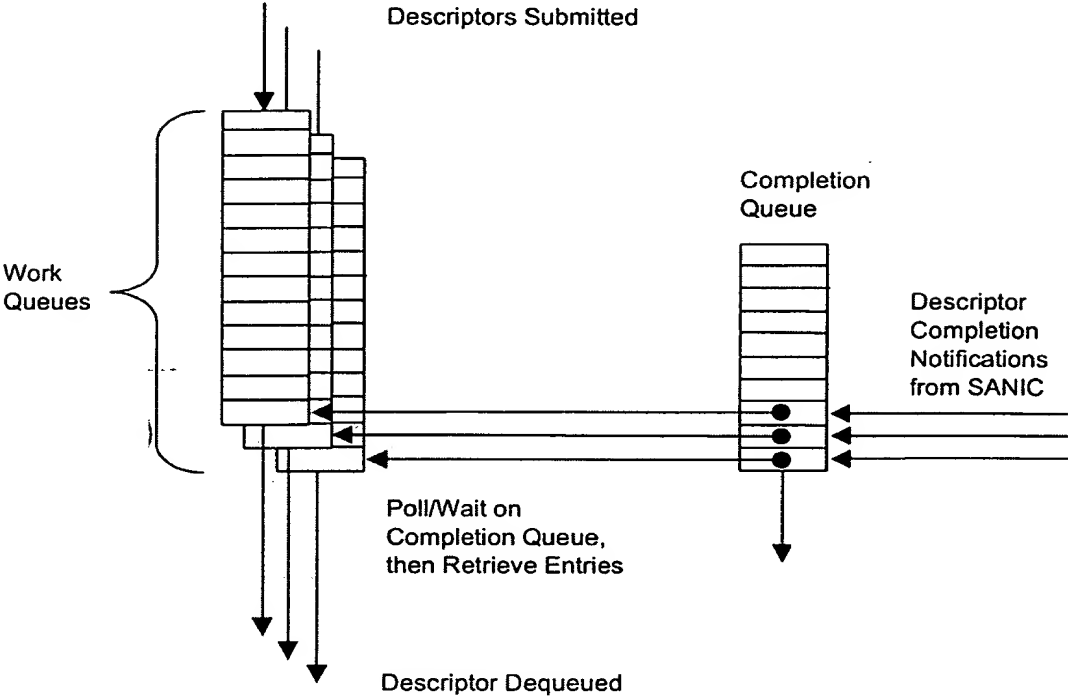


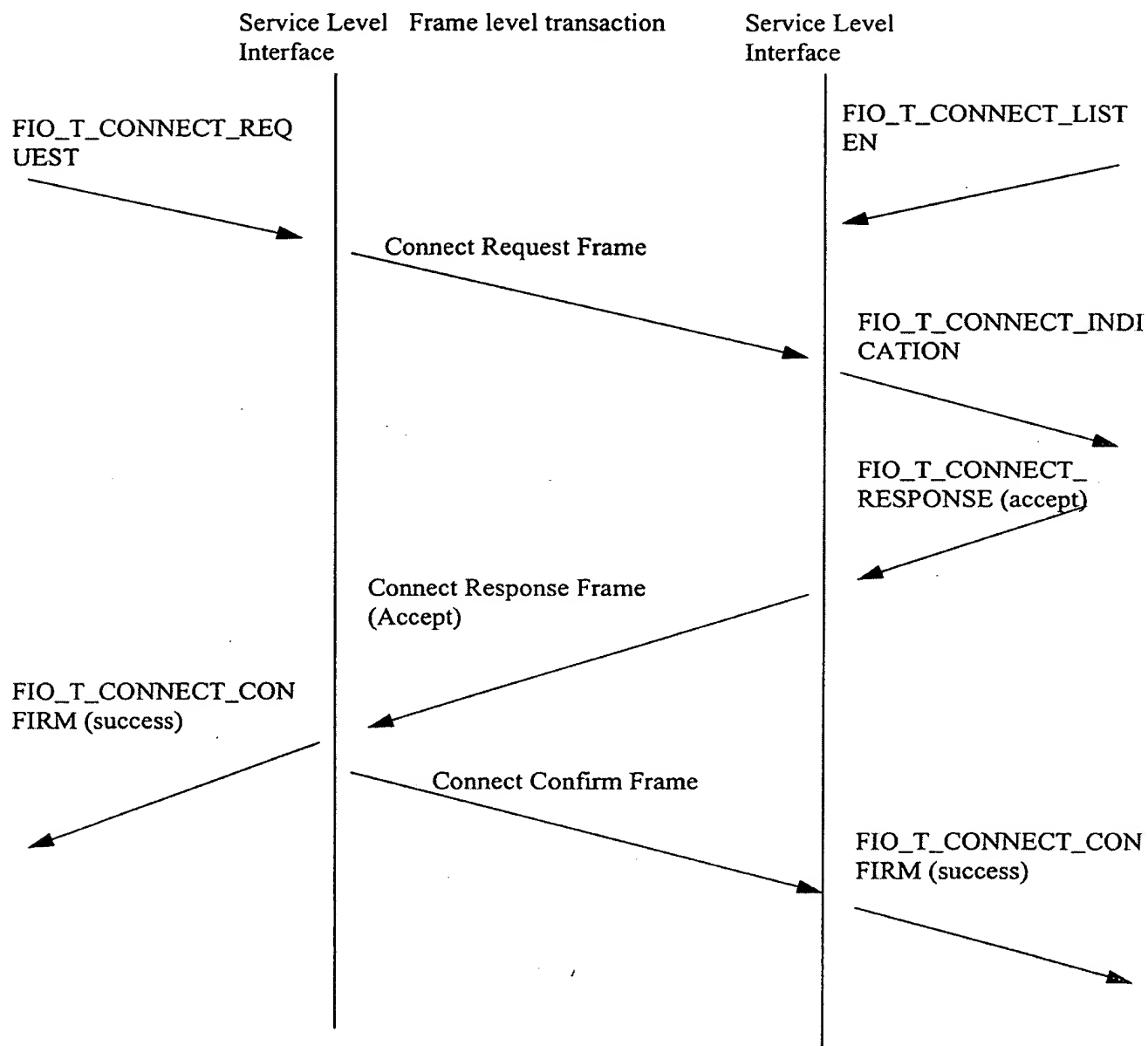
Figure 43 Connectionless QPs for UnReliable Datagram Operation

Figure 45 Completion queue model



004T60" 172T9960

**Figure 46 Connection establishment accept
frame transmission sequence**



09661214.091400

Service Level Interface	Frame level transaction	Service Level Interface
<p>1. Request</p> <p>2. Response</p>	<p>1. Request</p> <p>2. Response</p>	<p>1. Request</p> <p>2. Response</p>

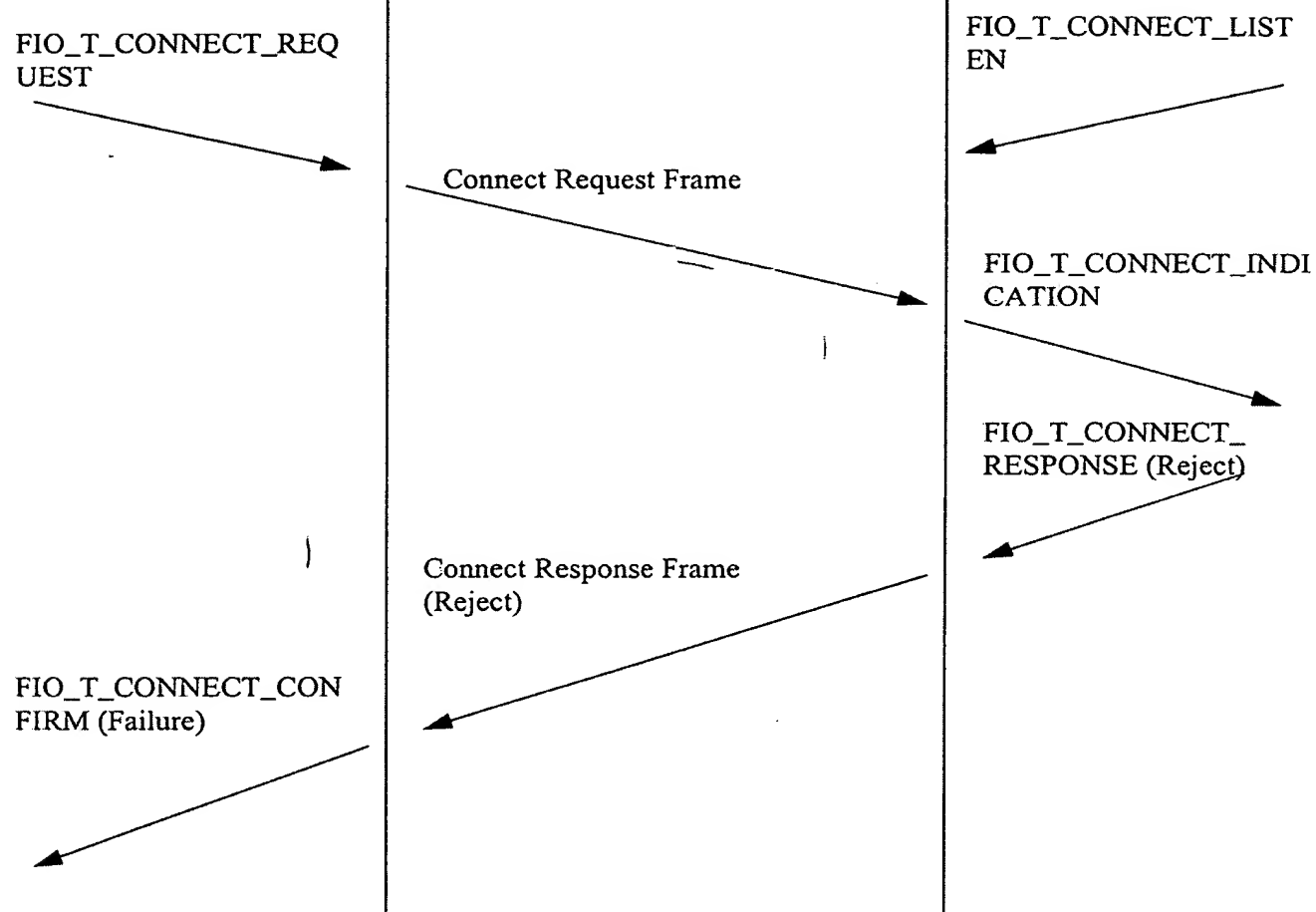
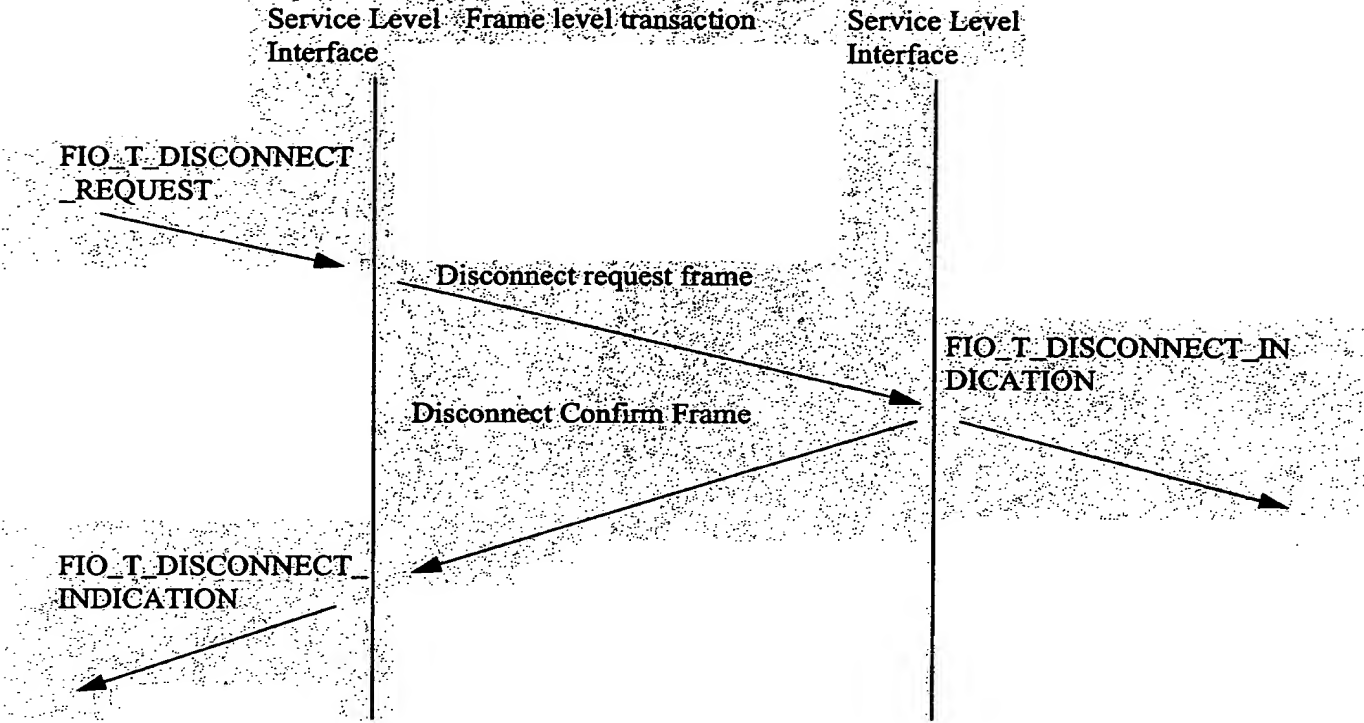
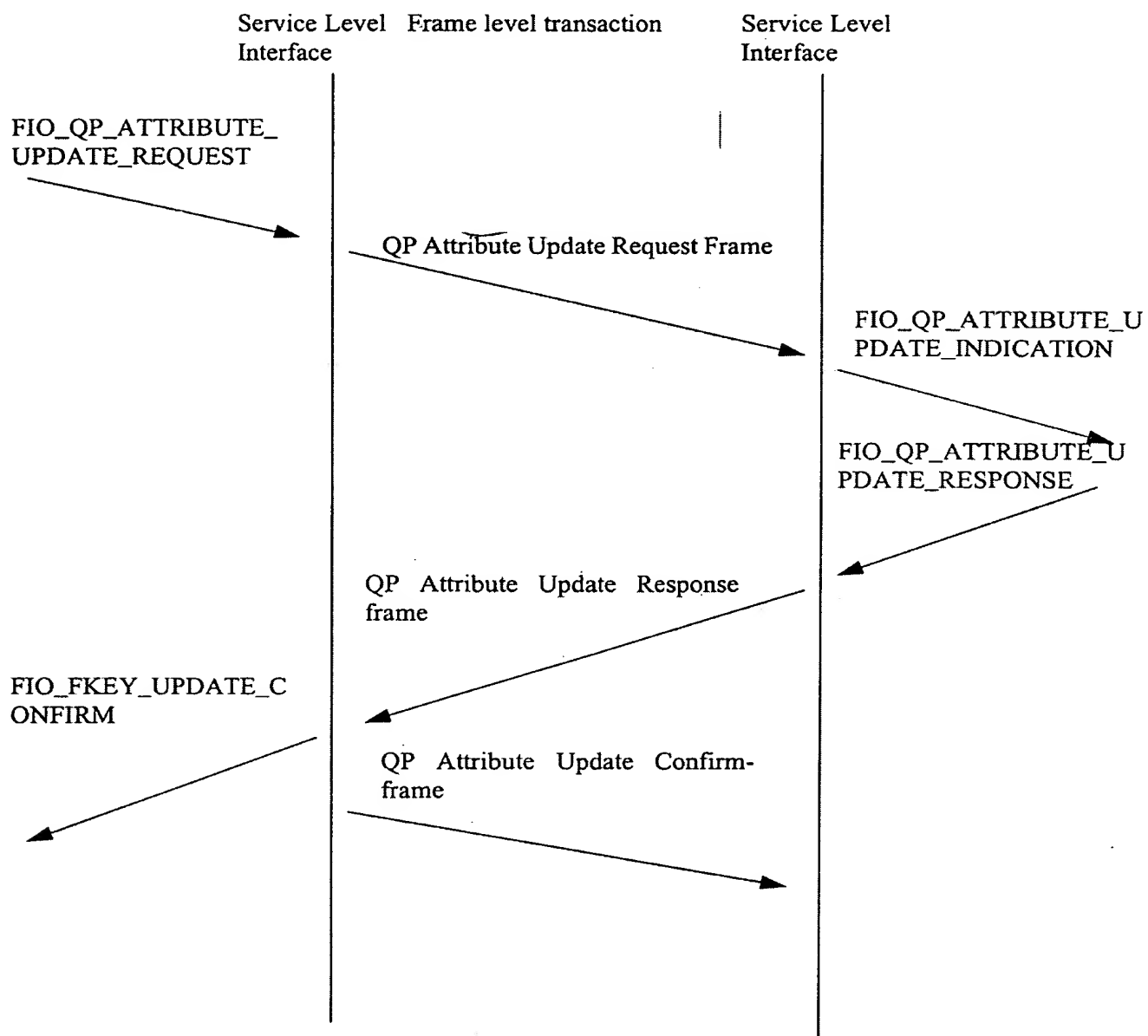


Figure 48 Connection teardown frame transmission sequence



004T60"4T2T9960

Figure 49 QP Attribute Update frame transmission sequence



004T60"42T960

Figure 50 - Port States in Spanning Tree

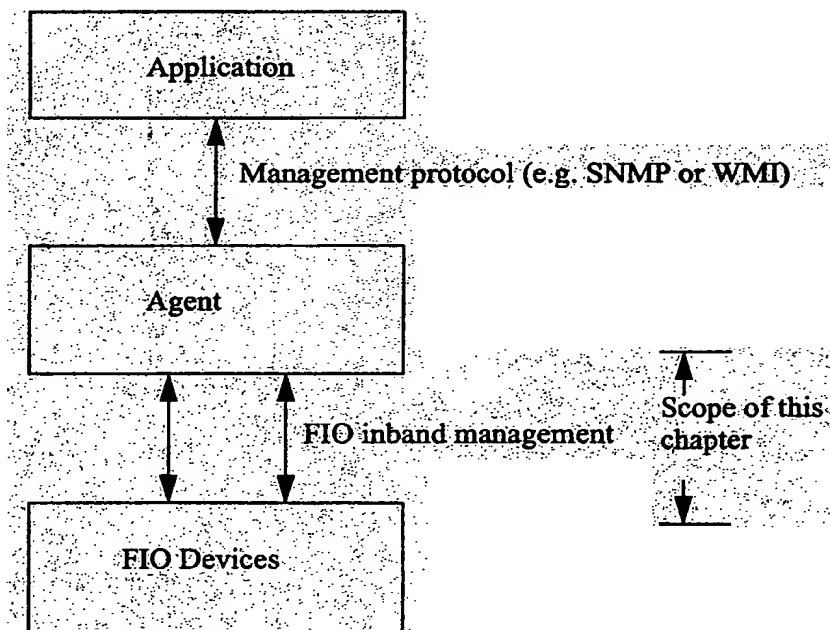
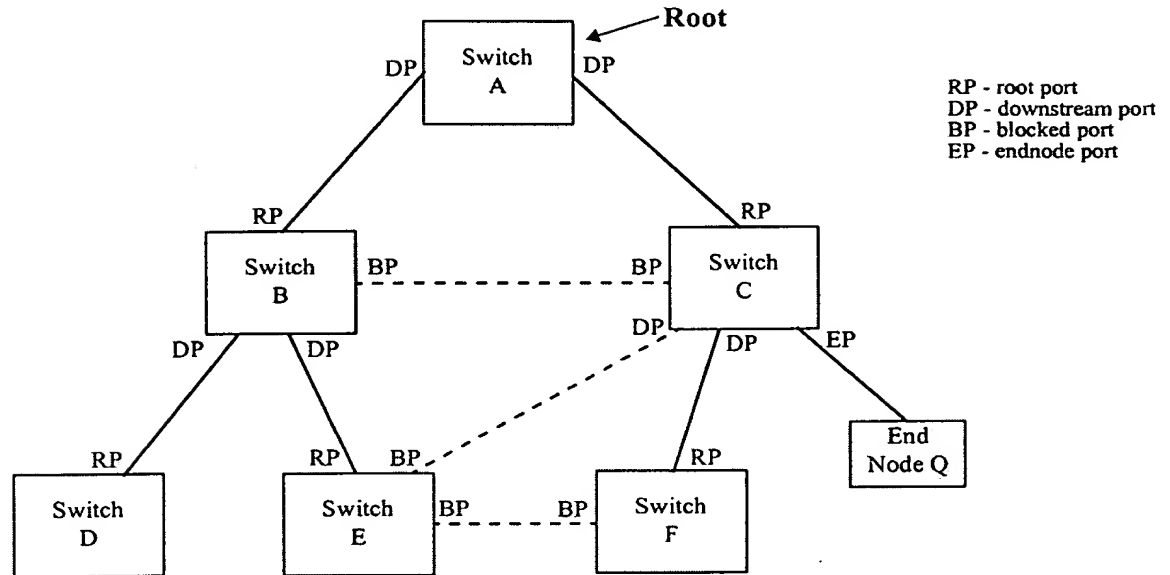
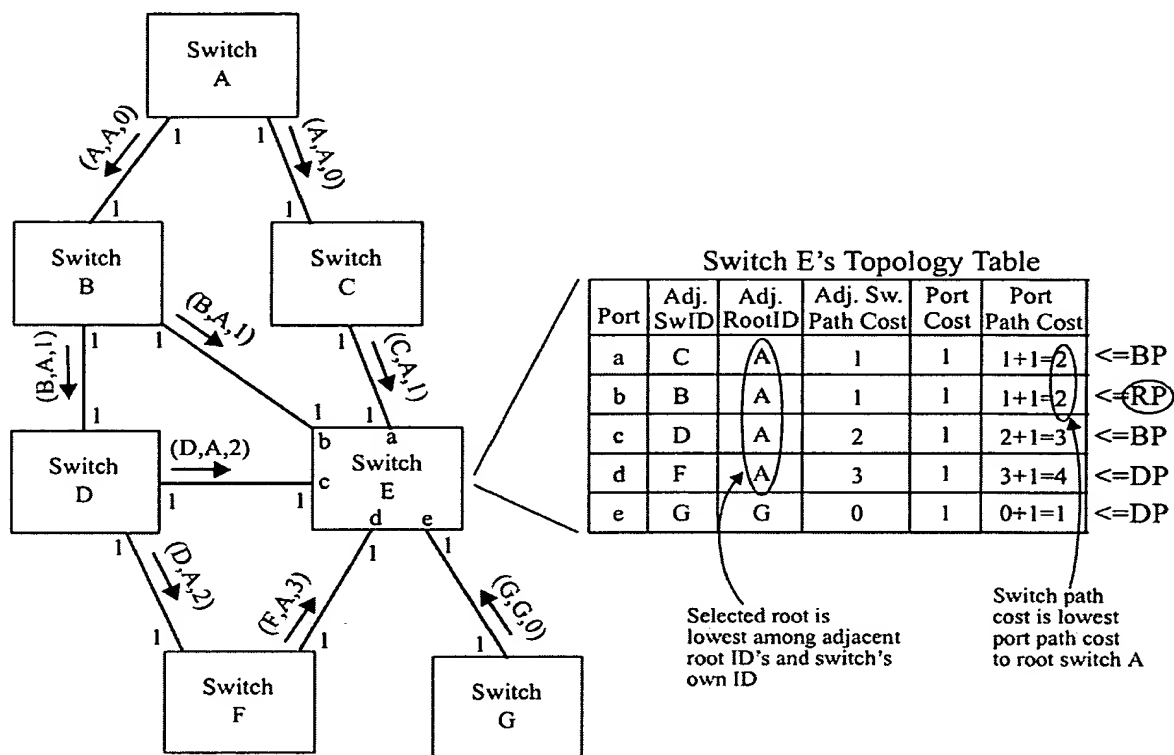


Figure 55

Figure 51 - Path Costs to Root (seen by Switch E)



The numbers beside each port indicate the port cost (i.e. path cost adder) assigned to that port.
 The (x,x,n) represents the NX message and 3 of its fields: switch ID, root ID, and path cost.

Figure 52 - Bundles

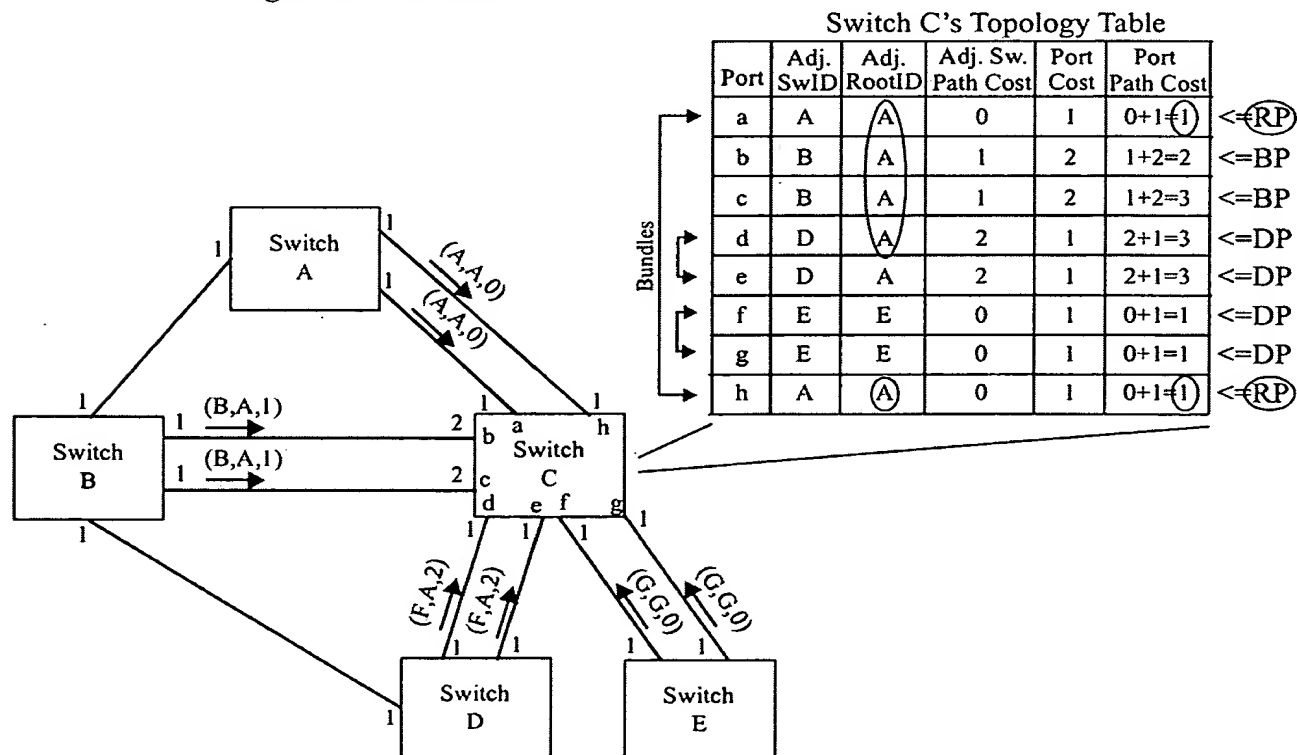
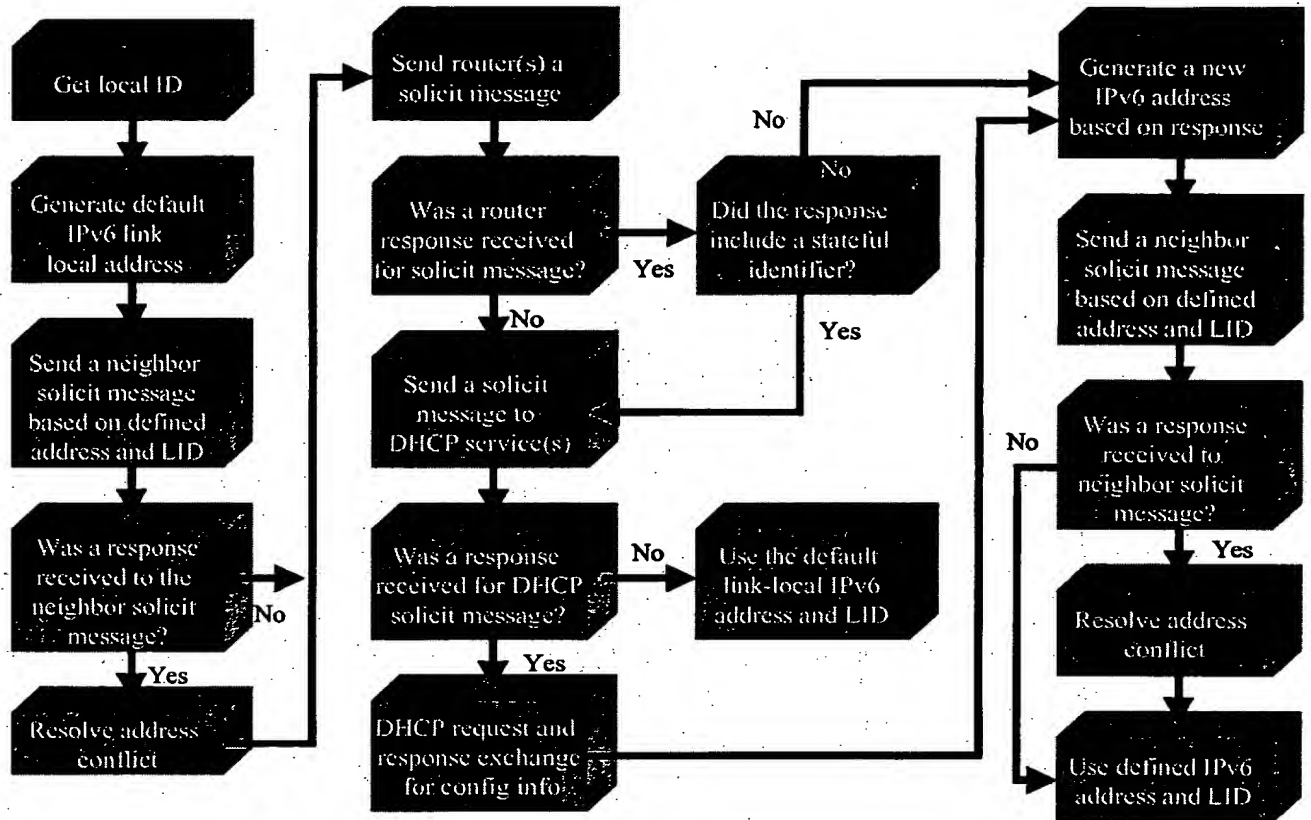


Figure 53 Node Configuration and Discovery



I/O Leaf Ports/States

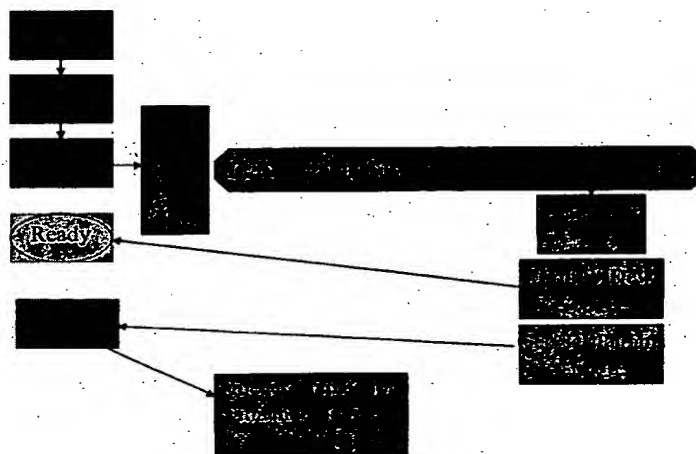


Figure 54 Boot Process

004T60"42T9960

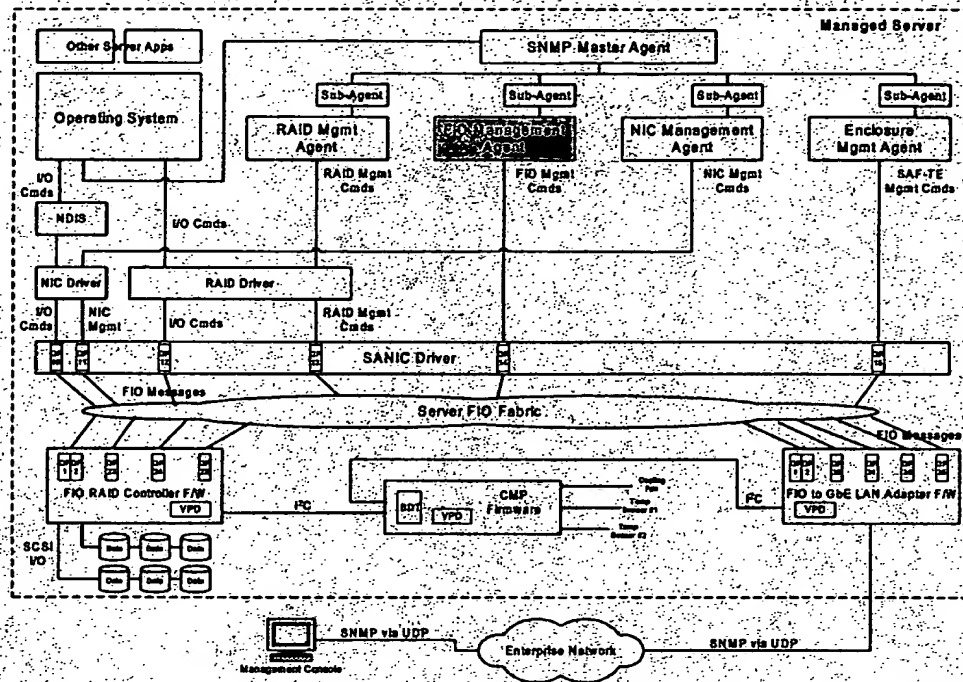


Figure 56

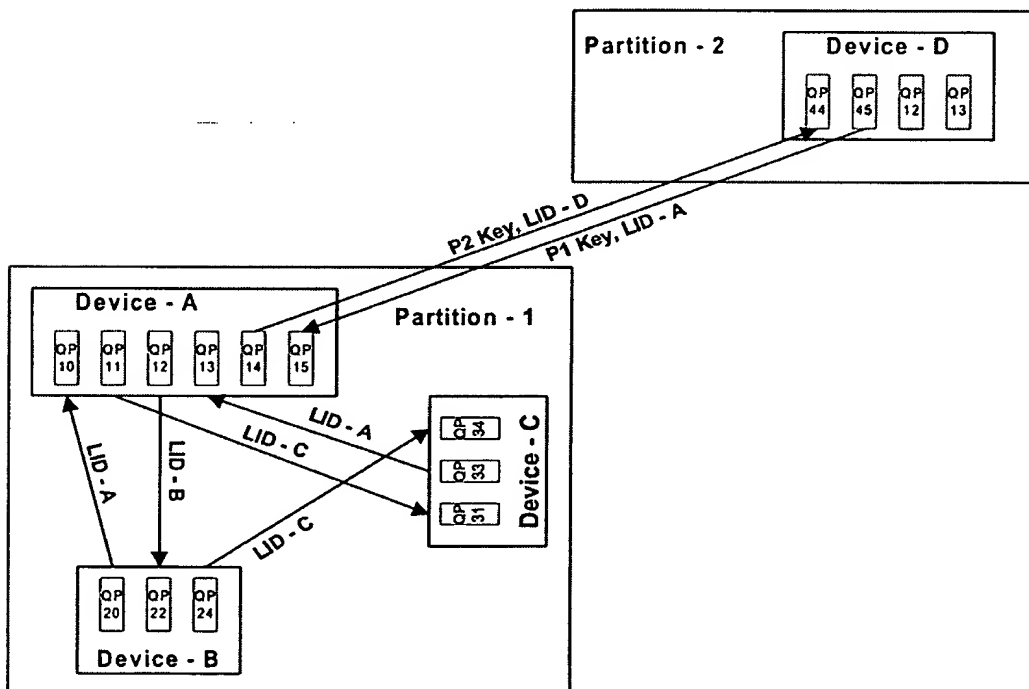


Figure 59

004760-4727960

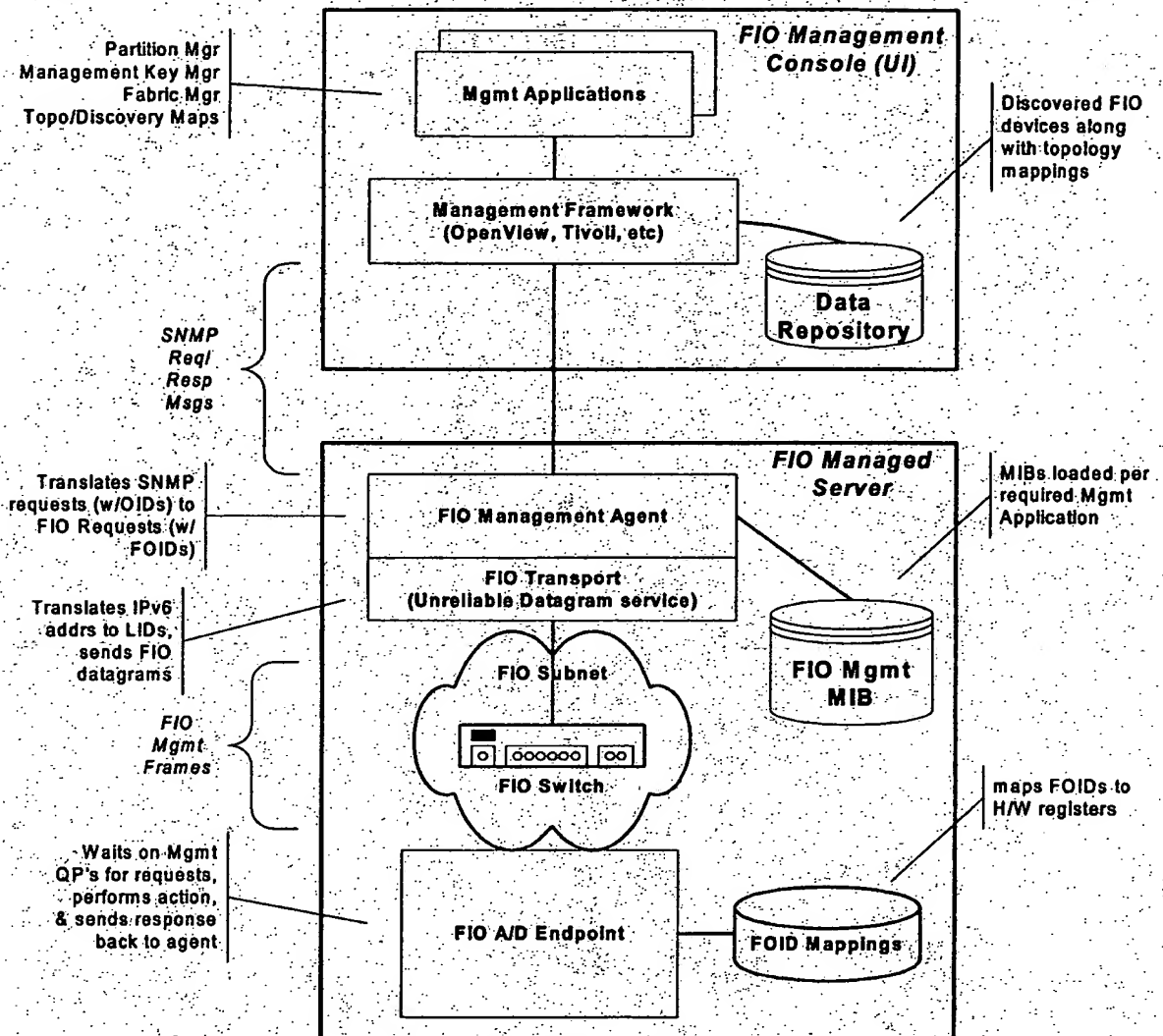


Figure 57

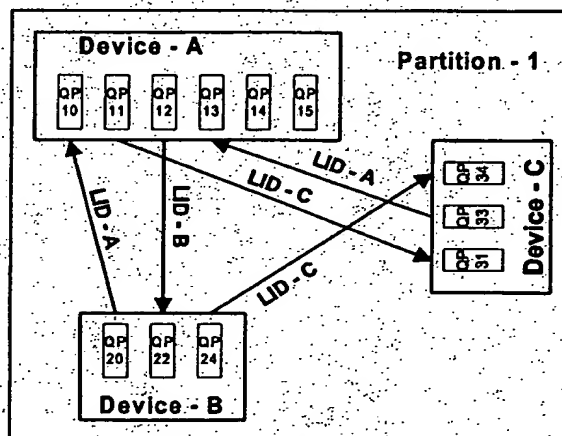


Figure 58

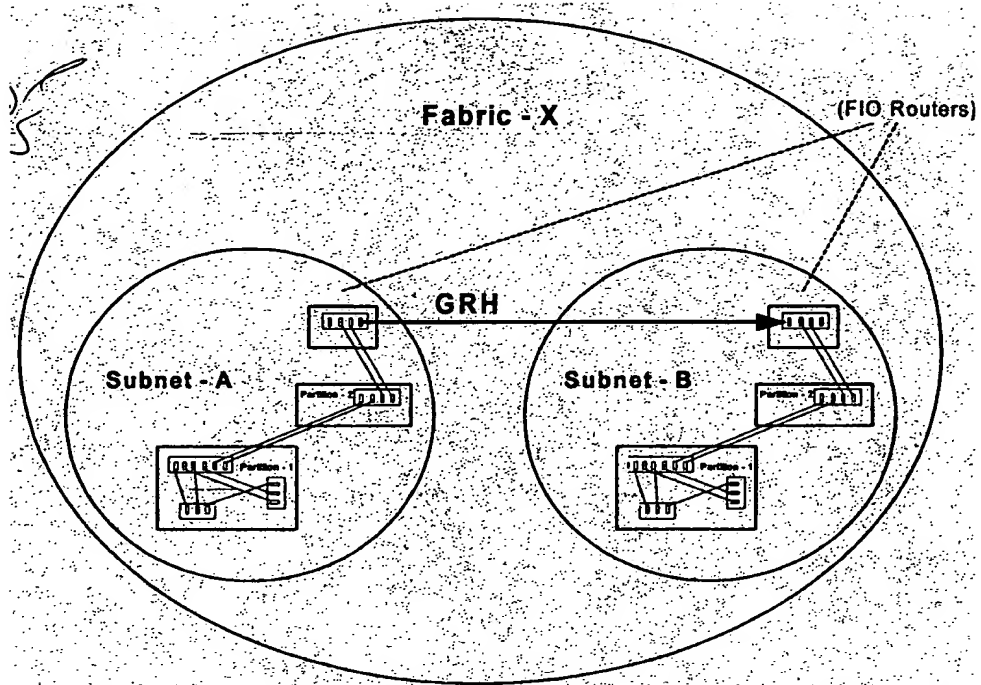
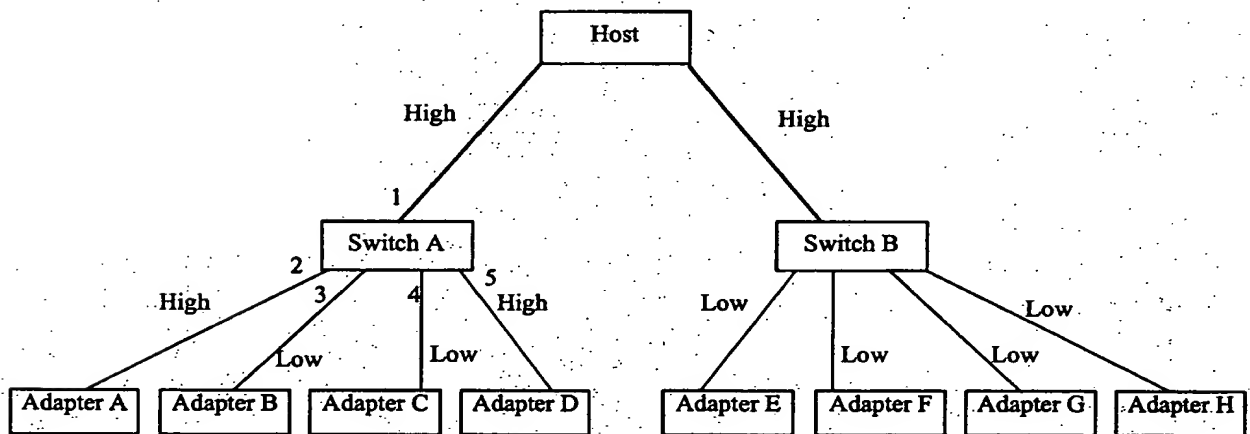
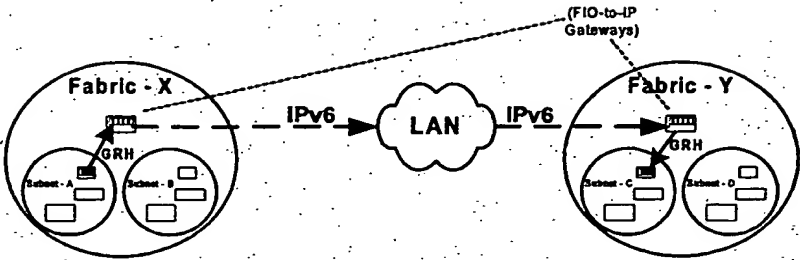


Figure 60

Figure 63 Simple Tree with Mixed Bandwidth Links and Adapter Leaves





While IPv6 interoperability is a key advantage for FIO, management messages may not route via IP

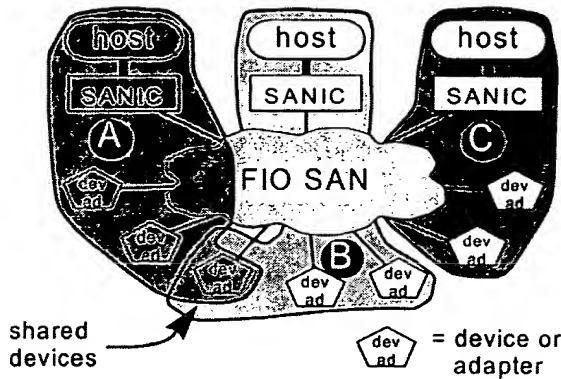


Figure 62 Example Endpoint Partitions of an FIO-Connected Cluster

Figure 65 - Simple Tree with Mixed Bandwidth Links and Adapter and Router Leaves

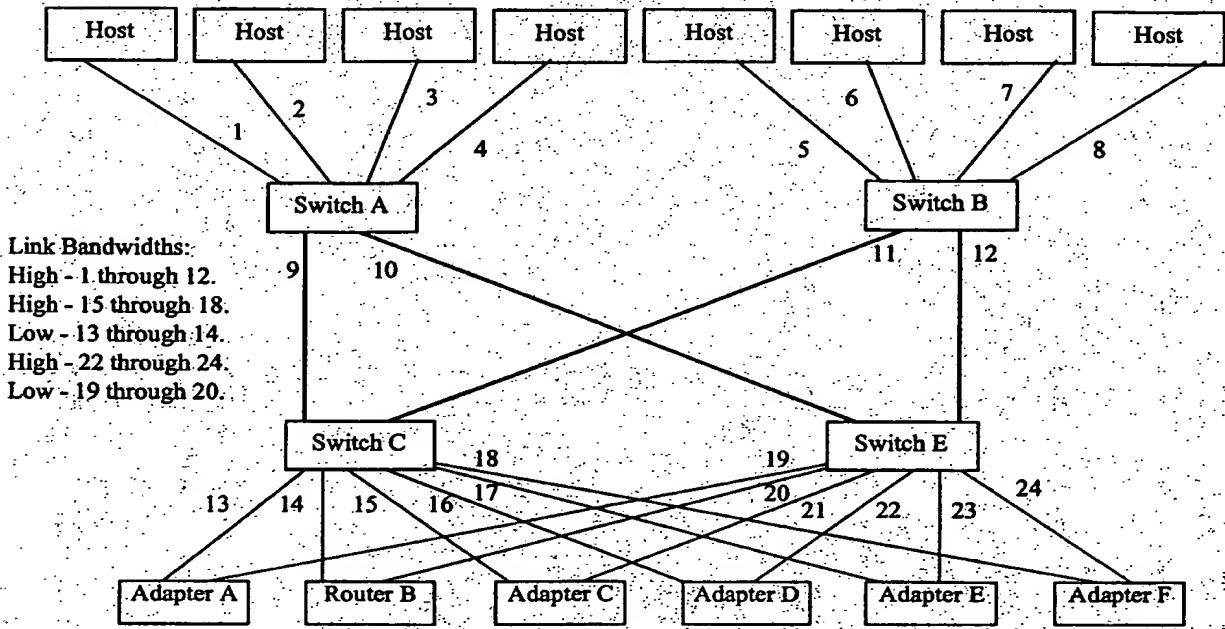
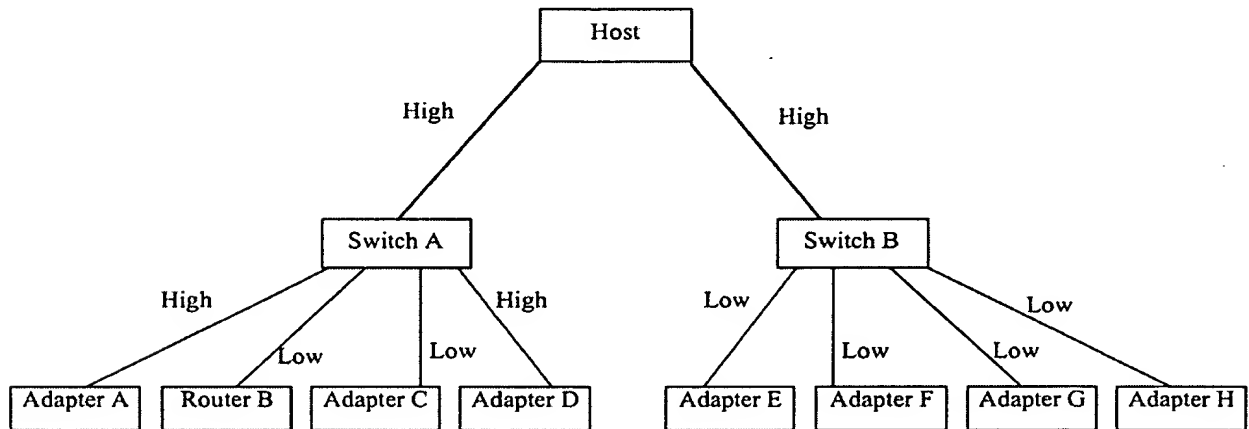


Figure 64 Simple Tree with Mixed Bandwidth Links and Adapter and Router Leaves



0961214-091400
004T50"4T2T950

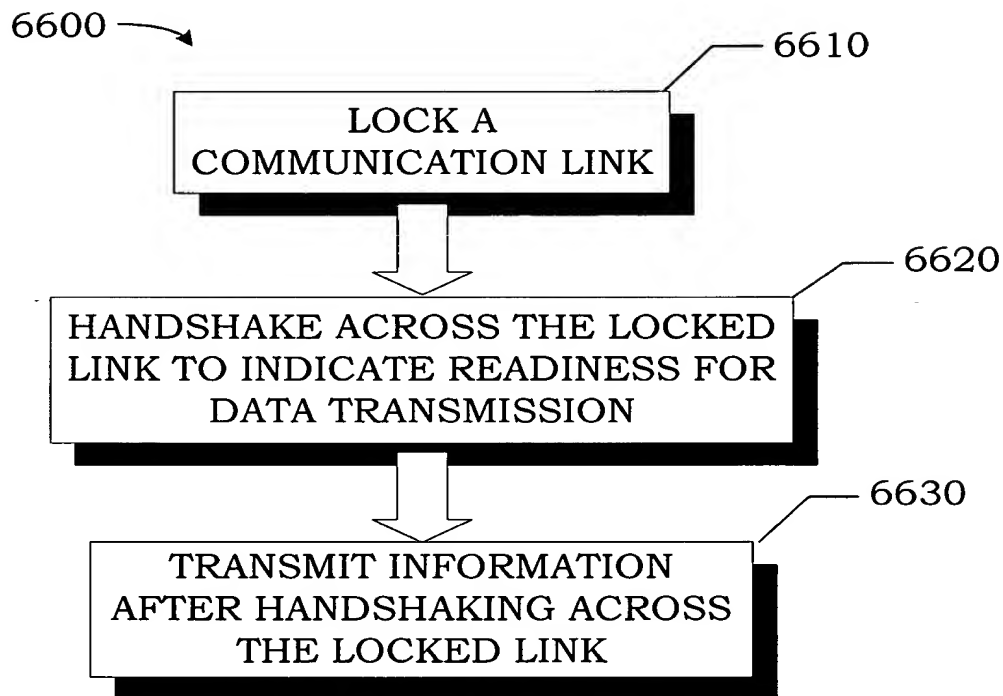


FIG. 66

004T60"42T960

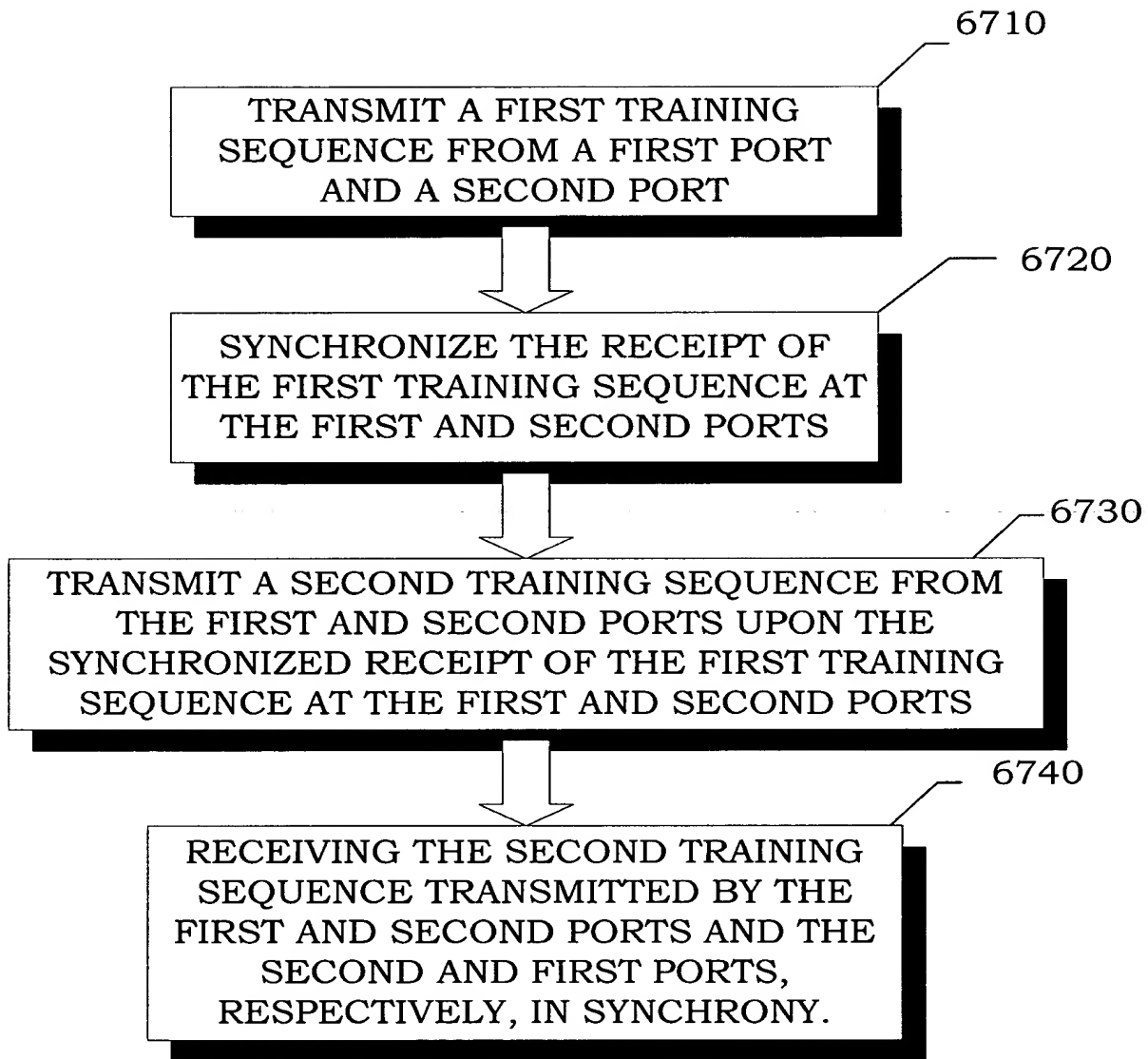


FIG. 67

004760" HT 27 9160

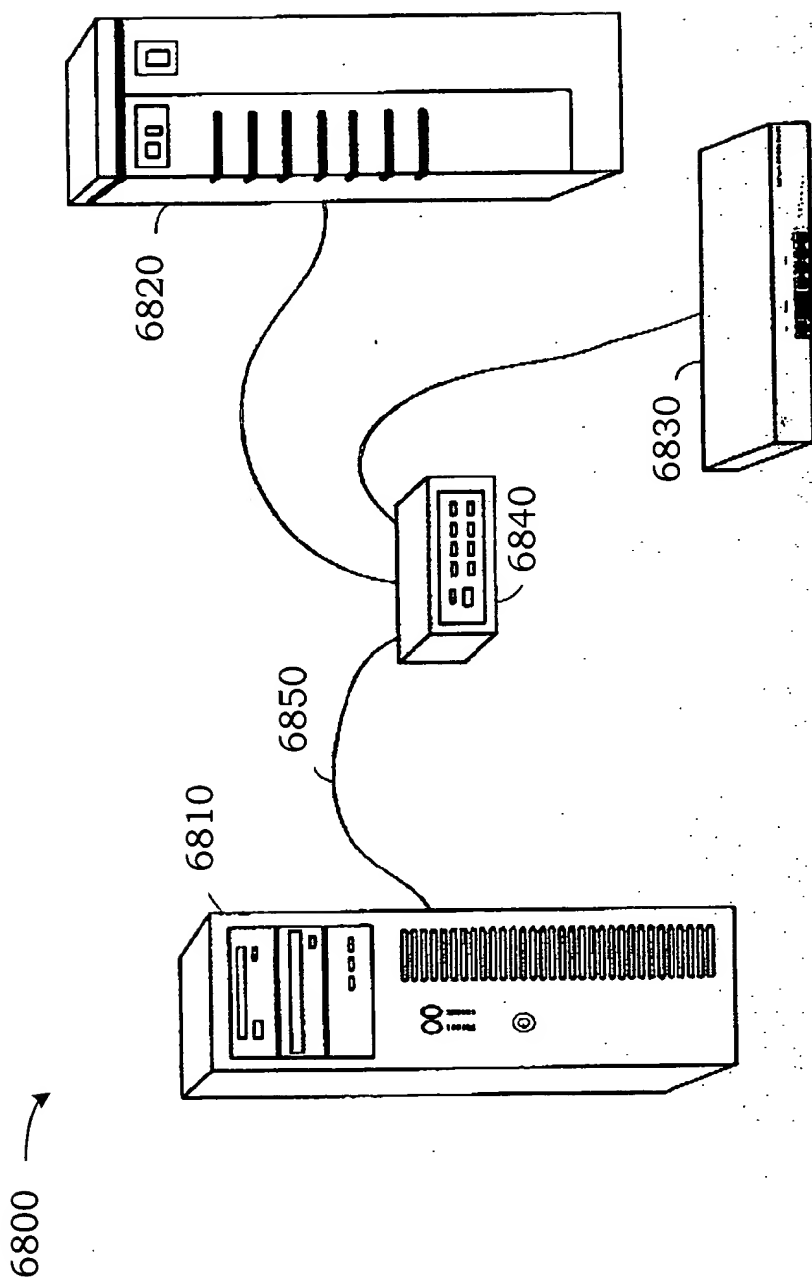


FIG. 68

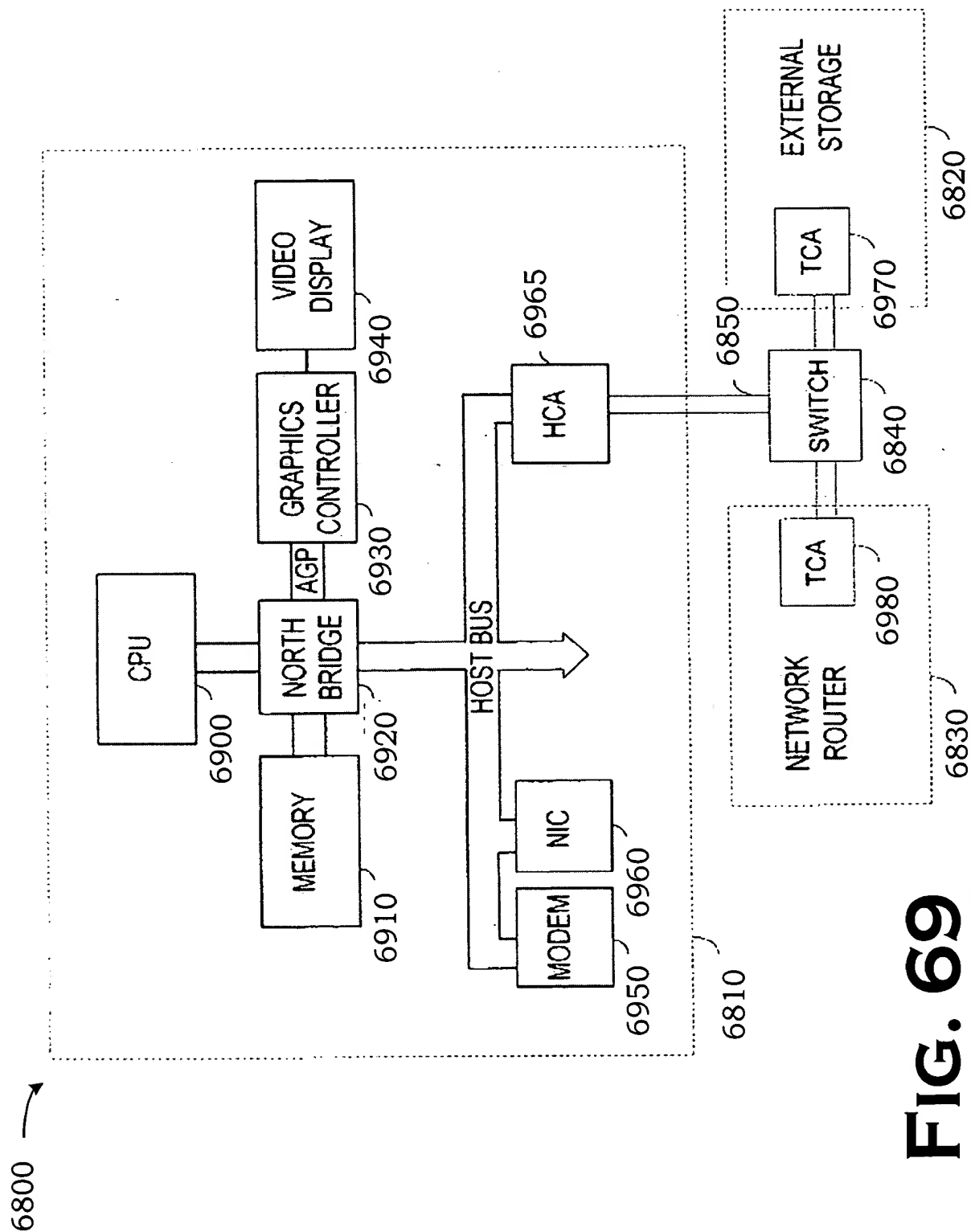


Fig. 6

004150"41219360

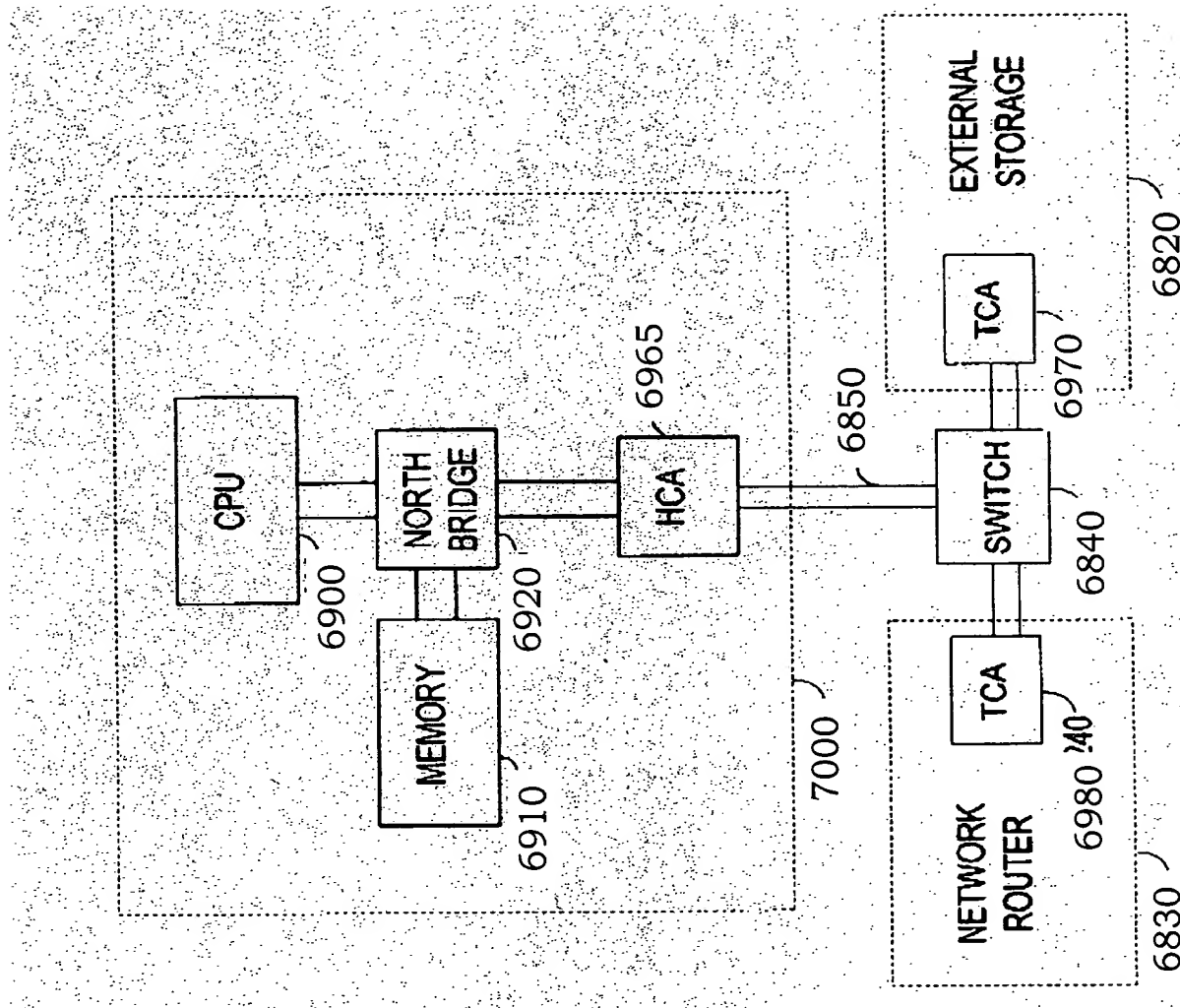


FIG. 70

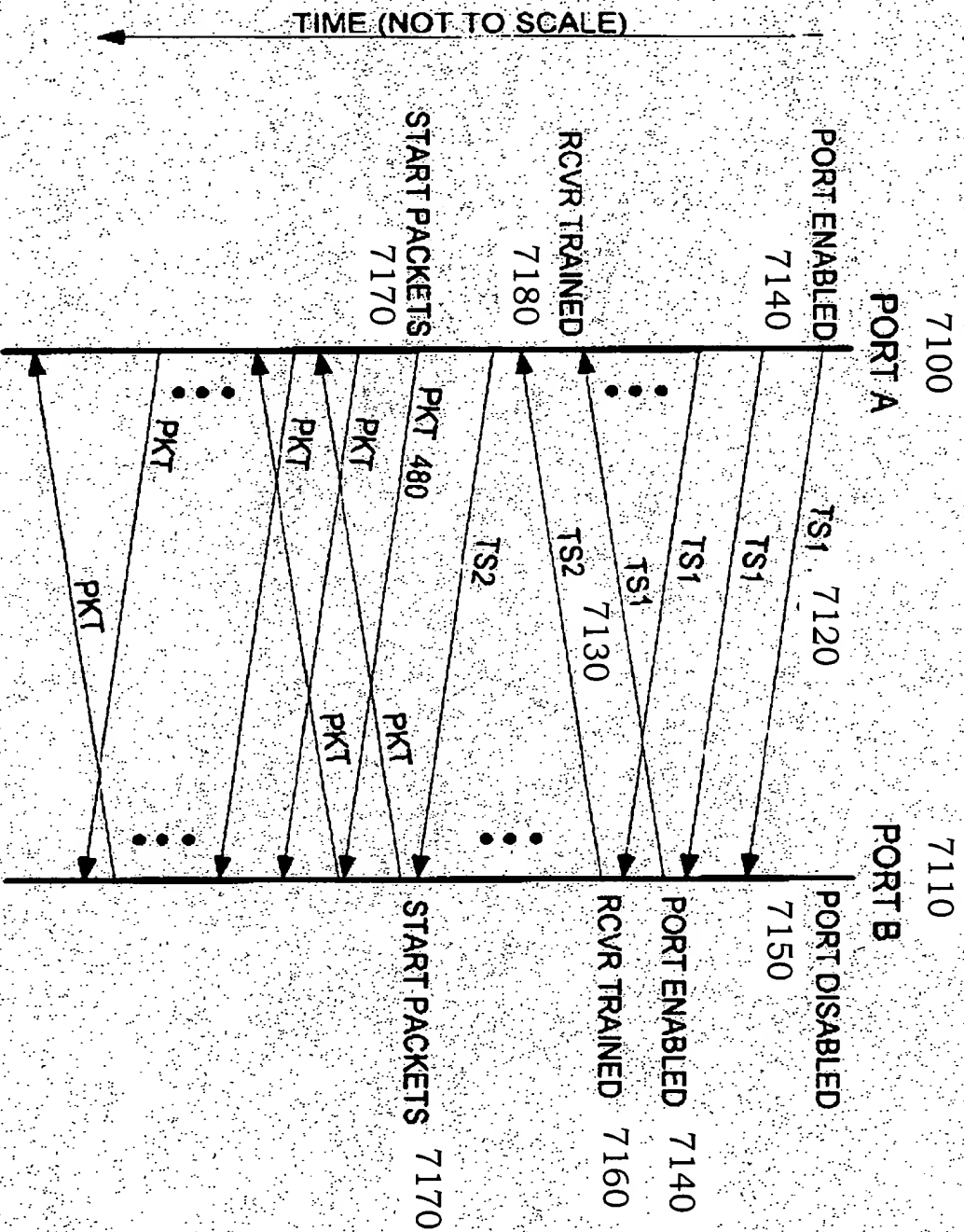


FIG. 71

09661214-091400

00000000000000000000

Physical Link Lane Identifiers

LANE IDENTIFIER	8B/10B CODE NAME	NEGATIVE RD	POSITIVE RD
LID 0	D0.0	10011 10100	01100 01011
LID 1	D1.0	01110 10100	10001 01011
LID 2	D2.0	10110 10100	01001 01011
LID 3	D4.0	11010 10100	00101 01011
LID 4	D8.0	11100 10100	00011 01011
LID 5	D15.0	01011 10100	10100 01011
LID 6	D16.0	01101 10100	10010 01011
LID 7	D23.0	11101 00100	00010 11011
LID 8	D24.0	11001 10100	00110 01011
LID 9	D27.0	11011 00100	00100 11011
LID 10	D29.0	10111 00100	01000 11011
LID 11	D30.0	01111 00100	10000 11011

FIG. 73

00460" 422950

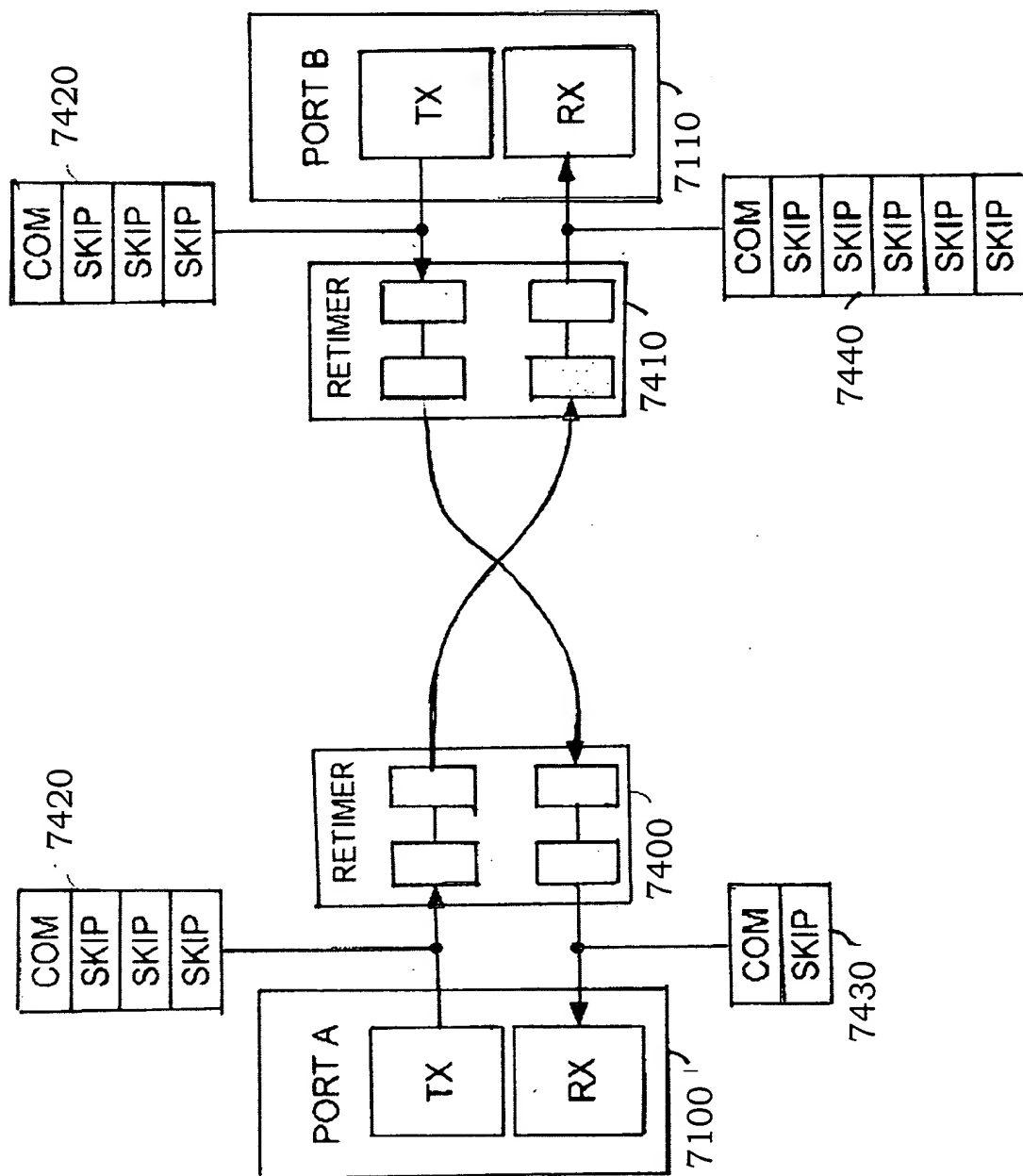


FIG. 74

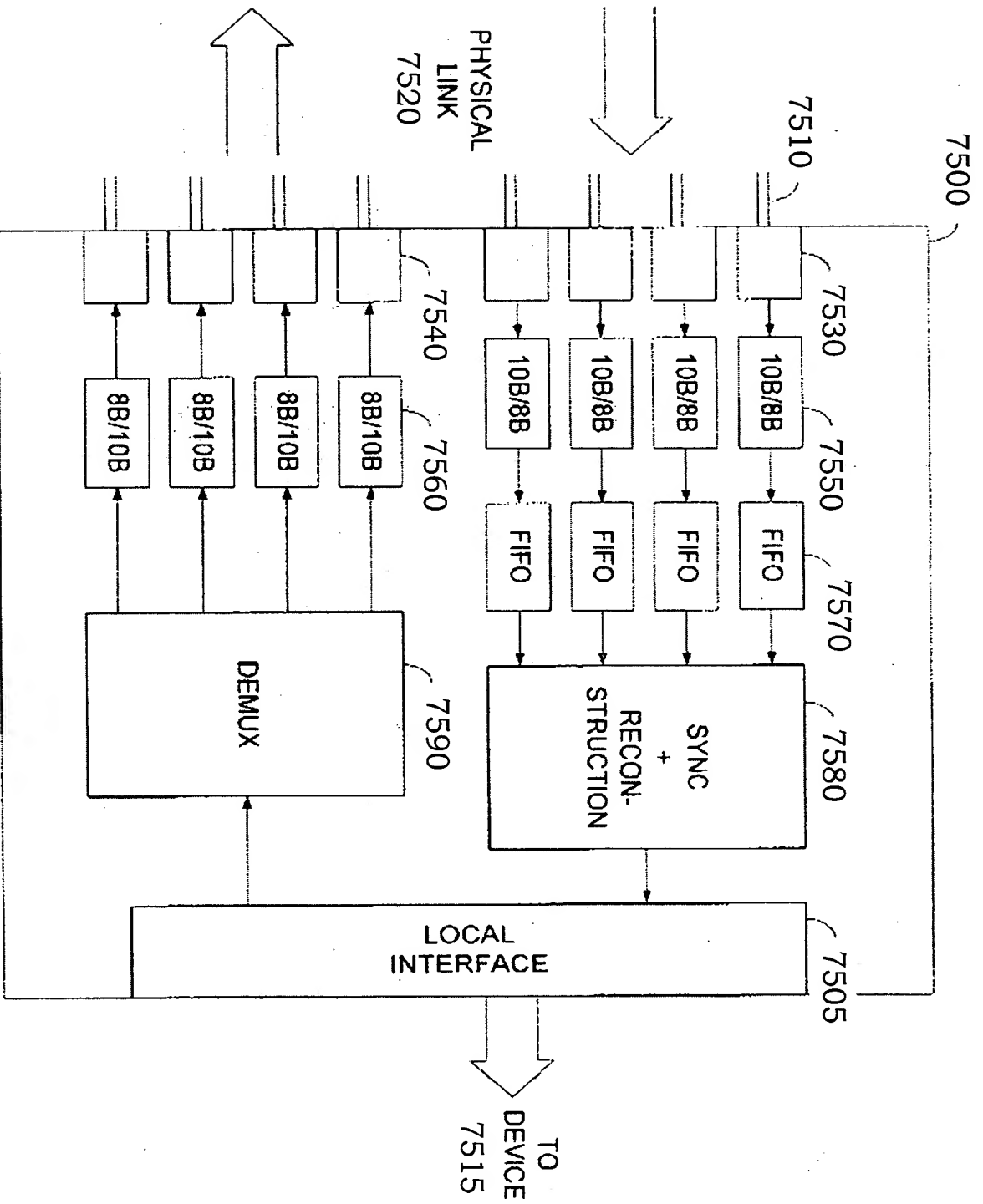
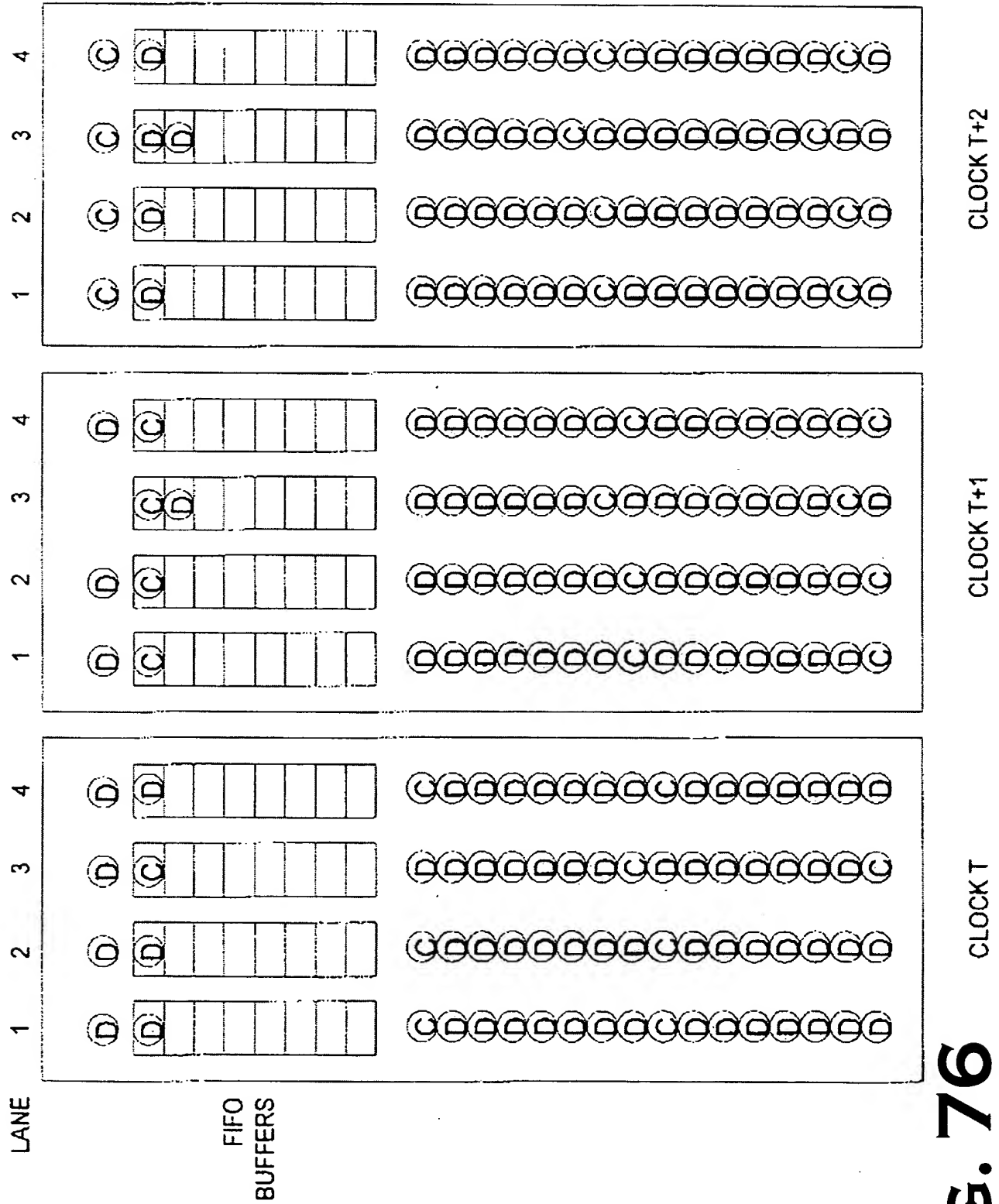


FIG. 75

09661214.091400

004F60"4F2F960



7700 →

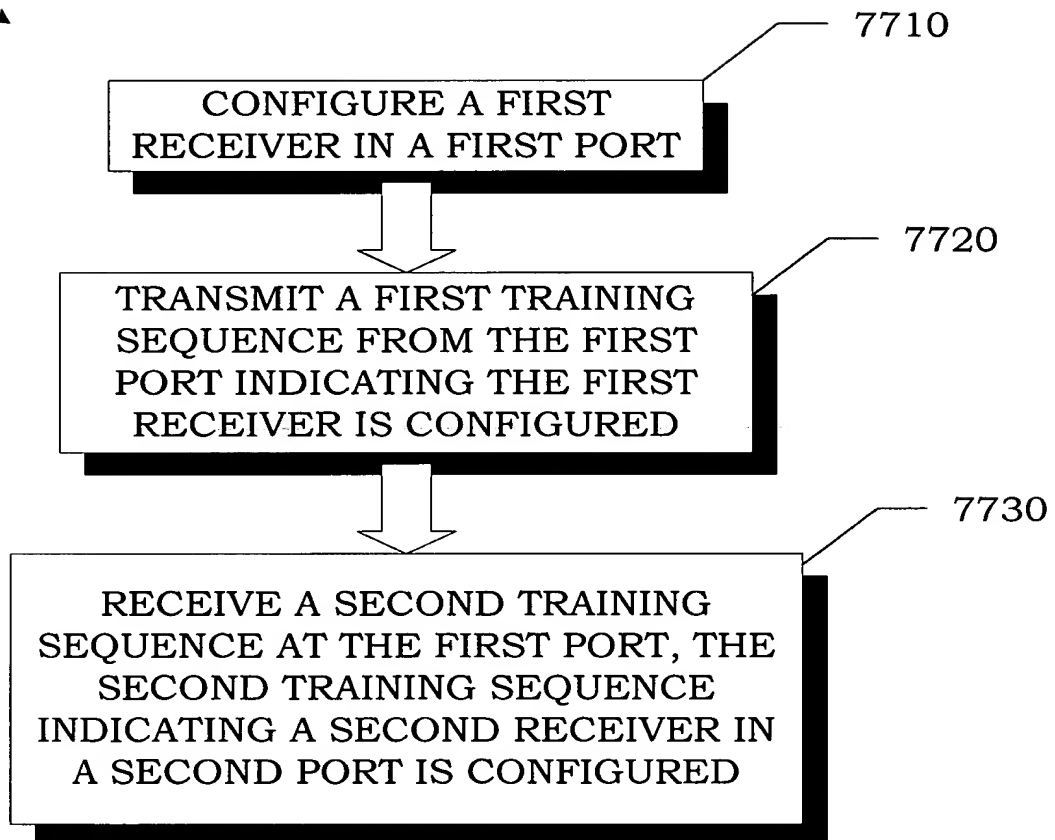


FIG. 77

09661214-091400